Domeniul Inginerie si Management

TEZĂ DE DOCTORAT

STUDII ȘI CERCETĂRI CU PRIVIRE LA MANAGEMENTUL PRODUCȚIEI DE SERIE

Student-doctorand: **Mircea Teodorescu**

Conducător științific: **Prof. Dr. Ing. Florin Lungu**

Comisia de evaluare a tezei de doctorat:

Preşedinte: Prof. Dr. Inf, Ec. **Emilia Ciupan** - Universitatea Tehnică din Cluj-Napoca;

Conducător ştiinţific: Prof. Dr. Ing. **Florin Lungu** - Universitatea Tehnică din Cluj-Napoca;

Referenţi:
- Prof. Dr. Ing. **Liviu Moldovan** - Universitatea de Medicină, Farmacie, Știință și Tehnologie "G.E. Palade" Târgu-Mureș;
- Prof. Dr. Ing. **Gabriela Proștean**- Universitatea Politehnica din Timișoara;
- Prof. Dr. Ing. **Nicolae Ungureanu** - Universitatea Tehnică din Cluj-Napoca.

– Cluj-Napoca –
2024

MULȚUMIRI

Cu profundă recunoștință și considerație, doresc să aduc mulțumiri deosebite conducătorului științific al acestei teze de doctorat, domnului Prof. dr. ing. Florin Lungu. Fără sprijinul său constant, această lucrare nu ar fi putut ajunge la forma sa actuală. Îndrumarea sa atentă, expertiza sa academică și profesională, precum și încrederea și încurajările oferite pe parcursul întregii perioade de cercetare, au fost esențiale pentru succesul acestei lucrări. Domniei sale îi adresez cele mai sincere mulțumiri pentru implicarea sa continuă și pentru că mi-a fost un mentor dedicat și un model de excelență academică și profesională.

De asemenea, doresc să îmi exprim profunda recunoștință față de membrii comisiei de îndrumare, Prof. dr. ing. Ec. Laura Bacali, Prof. dr. ing. Adrian Pâslă, pentru contribuțiile lor valoroase și pentru suportul lor oferind observații riguroase și pertinente, dar și sprijin academic pe tot parcursul desfășurării lucrării. Aprecierea mea față de dedicația lor este imensă, iar contribuțiile lor au avut un impact decisiv asupra acestei teze.

Doresc să adresez mulțumiri sincere domnului Conf. dr. ing. Radu Vlad pentru timpul acordat, expertiza oferită și sprijinul tehnic esențiale în depășirea provocărilor și în îmbunătățirea calității rezultatelor.

Mulțumirile mele se îndreaptă, de asemenea, către toți cei care, într-un fel sau altul, au contribuit la realizarea acestei lucrări. Fie că sprijinul lor a fost direct sau indirect, recunoștința mea față de aportul lor este profundă și sinceră. Implicarea lor a făcut posibilă finalizarea cu succes a acestei cercetări, iar acest sprijin va rămâne neuitat.

În final, doresc să aduc mulțumiri speciale familiei mele, care m-a susținut necondiționat pe parcursul acestui lung și provocator proces. Sprijinul lor moral, încurajările constante și înțelegerea lor au fost fundamentale pentru reușita mea. Această realizare nu ar fi fost posibilă fără sprijinul lor neprețuit, pentru care le sunt profund recunoscător

Mulțumesc bunului Dumnezeu pentru puterea de a duce la bun sfârșit această cercetare!

CUPRINSUL TEZEI

INTRODUCERE

a. Motivația și necesitatea demersului (Argument)

În contextul actual al pieței globalizate, optimizarea programării producției nu este doar o necesitate, ci și un imperativ pentru companiile care doresc să rămână competitive. Creșterea complexității lanțurilor de aprovizionare, diversificarea cerințelor clienților și presiunea constantă pentru reducerea costurilor impun adoptarea unor soluții tehnologice avansate. În acest sens, inteligența artificială (IA) și algoritmii de optimizare oferă un potențial imens pentru îmbunătățirea eficienței operaționale. Utilizarea acestor tehnologii permite companiilor să planifice mai precis, să reducă ciclurile de producție și să minimizeze erorile, contribuind astfel la îmbunătățirea calității produselor și la creșterea profitabilității.

În plus, în era digitalizării, adoptarea IA în managementul producției devine un factor diferențiator esențial. Companiile care întârzie să implementeze astfel de soluții riscă să rămână în urmă, pierzându-și avantajul competitiv. Prin urmare, demersul de cercetare și aplicare a algoritmilor de optimizare în programarea producției nu este doar relevant, ci și esențial pentru asigurarea sustenabilității și succesului pe termen lung. Aceasta nu doar că sprijină îmbunătățirea performanței interne, dar și contribuie la schimbările rapide ale pieței, asigurând astfel viitorul organizațiilor în mediul de afaceri dinamic de astăzi.

b. Actualitatea și importanța temei

Tema optimizării programării producției prin utilizarea algoritmilor de IA este de o actualitate incontestabilă în contextul transformării digitale care domină industria contemporană. Într-o perioadă în care inovația tehnologică redefinește procesele de producție, adoptarea acestor metode devine esențială pentru a face față provocărilor economice și operaționale. Algoritmii de optimizare, inclusiv cei genetici și hibrizi, oferă soluții eficiente pentru gestionarea complexității proceselor de producție, reducând timpii de execuție, minimalizând resursele utilizate și îmbunătățind calitatea produselor.

Importanța acestei teme rezidă nu doar în capacitatea de a spori competitivitatea companiilor, ci și în potențialul de a revoluționa managementul producției la nivel global. Pe măsură ce companiile adoptă tot mai mult IA pentru a-și optimiza procesele, capacitatea de a implementa și integra aceste tehnologii devine un criteriu crucial pentru succesul pe termen lung. Astfel, cercetarea în acest domeniu contribuie semnificativ la dezvoltarea unei economii bazate pe eficiență și inovație, consolidând importanța acestei teme în peisajul actual al industriei.

c. Obiectivele tezei

Cercetarea întreprinsă în cadrul tezei are ca scop principal investigarea și evaluarea aplicării algoritmilor de inteligență artificială (IA), în special a celor genetici și hibrizi, în optimizarea programării producției de serie. Având în vedere complexitatea și dinamismul mediului industrial actual, teza își propune să atingă următoarele obiective:

1. Identificarea elementelor inovative care contribuie la îmbunătățirea procesului de planificare și programare a producției de serie. Acest obiectiv vizează descoperirea și analiza metodelor noi, emergente în domeniul programării producției, cu accent pe aplicațiile IA.
2. Optimizarea proceselor de producție prin implementarea tehnicilor avansate de inteligență artificială, în special algoritmi genetici și hibrizi. Se urmărește investigarea modului în care aceste tehnici pot aduce îmbunătățiri în eficiența și exactitatea programării producției.
3. Evaluarea aplicabilității IA în managementul producției prin analizarea modului în care diverse tipuri de algoritmi IA pot fi aplicate pentru a crește eficiența și adaptabilitatea proceselor de producție.
4. Identificarea principalelor tendințe în managementul programării producției de serie. Acest obiectiv vizează studierea și înțelegerea evoluției recente și a direcțiilor viitoare în acest domeniu, pentru a propune soluții relevante și actuale.

Aceste obiective sunt esențiale pentru a oferi o imagine clară asupra impactului și potențialului tehnologiilor IA în optimizarea proceselor de producție, asigurând astfel o contribuție semnificativă la dezvoltarea și eficientizarea managementului producției.

d. Structura (rezumat)

Teza este structurată în trei părți principale, fiecare organizată în capitole care urmăresc să atingă obiectivele cercetării.

Partea generală începe cu Capitolul 1, care oferă o privire de ansamblu asupra fundamentelor managementului producției, incluzând aspecte esențiale precum planificarea și programarea producției, diversitatea tipurilor de producție și inovațiile recente, precum impactul tehnologiei IoT și abordarea problemelor de tip Job Shop Schedule.

Capitolul 2 continuă cu un studiu comparativ al metodelor de programare a producției bazate pe algoritmi de inteligență artificială, analizând strategiile de planificare, tehnicile de ordonanțare și bazele teoretice ale IA.

Partea specifică se deschide cu Capitolul 3, care prezintă metodologia de cercetare aplicată și discută în detaliu modelarea programării producției de serie. Acest capitol include analiza interviurilor ghidate și un studiu de caz asupra companiei ABC, explorând integrarea și implementarea tehnologiilor IA, precum și comparația între metodele exacte și cele euristice.

Partea experimentală este abordată în Capitolul 4, care se concentrează pe comprimarea rezultatelor aplicării funcției obiectiv, discutând rezultatele obținute, analiza cauză-efect și evaluarea SWOT a acestora. Teza se încheie cu concluzii ce sintetizează principalele contribuții și impactul cercetării asupra domeniului.

e. Metodologia (întrebarea de bază a cercetării, ipotezele de cercetare și metodele de cercetare utilizate)

Metodologia acestei teze se bazează pe o abordare integrată care combină analiza teoretică cu cercetarea aplicată pentru a răspunde întrebării principale a cercetării: „În ce măsură contribuie aplicarea algoritmilor de inteligență artificială (genetici și hibrizi) și a altor tehnici de optimizare la îmbunătățirea procesului de programare a producției de serie?”

Pentru a aborda această întrebare, au fost formulate următoarele ipoteze de cercetare:

Ipoteza 1. Metodele exacte oferă rezultate superioare celor euristice în programarea producției de serie.

Ipoteza 2. Algoritmii genetici hibrizi oferă rezultate superioare celor genetici clasici în ceea ce privește exactitatea soluțiilor oferite în programarea producției de serie.

Ipoteza 3. Algoritmii genetici clasici oferă rezultate diferite și neconcludente în rulările succesive în programarea producției de serie.

Ipoteza 4. Aplicarea modelului Guinet în programarea producției va conduce la îmbunătățiri semnificative în reducerea ciclurilor de producție.

Ipoteza 5. Rezultatele aplicării algoritmilor hibrizi au șanse ca pe viitor să se apropie de rezultatele aplicării metodelor exacte în programarea producției de serie.

Pentru a testa aceste ipoteze, au fost utilizate diverse metode de cercetare. Analiza literaturii a oferit contextul teoretic necesar pentru înțelegerea și aplicarea tehnicilor de optimizare în programarea producției. Interviurile ghidate cu experți din industrie au fost esențiale pentru a colecta perspective practice și pentru a evalua percepțiile asupra aplicabilității și eficienței algoritmilor IA. Studiul de caz privind firma ABC a permis investigarea concretă a implementării acestor tehnologii, comparând rezultatele obținute prin metode exacte și euristice. Analiza datelor prin intermediul

tehnicilor computaționale și evaluarea rezultatelor folosind analiza SWOT au fost utilizate pentru a testa validitatea ipotezelor și pentru a trage concluzii relevante asupra impactului IA în optimizarea programării producției.

PARTEA GENERALĂ

CAPITOLUL 1: STUDIU PRIVIND STADIUL ACTUAL ÎN DOMENIUL MANAGEMENTULUI PRODUCȚIEI

1.1. Fundamentele managementului producției

Acest subcapitol rezumă aspectele cheie ale managementului producției, care reprezintă unul dintre pilonii operaționali ai oricărei organizații industriale. Concret, subcapitolul începe prin a identifica managementul producției ca definiție a proceselor de planificare, coordonare și control a activităților asociate transformării de resurse brute în produse finite. Prin urmare, managementul producției implică numeroase decizii strategice și operaționale, inclusiv selectarea materiei prime, planificarea producției, controlul calității și livrarea produselor.

În termeni istorici, managementul producției a evoluat în timp, începând cu Revoluția Industrială. Aceasta a reprezentat o tranziție de la producția umană la producția mecanizată și automatizată, care a oferit fundația pentru conceptele moderne de producție. Ulterior, de-a lungul secolului XX, noile teorii și practici au fost dezvoltate, inclusiv sistemul de producție Toyota și conceptul de producție „*lean manufacturing*”, care se concentrează pe efectuarea eficientă a muncii și eliminarea risipei. [1]

Elementele principale ale funcțiilor managementului producției sunt (printre altele): planificarea și controlul proceselor. Acestea includ stabilirea programelor de producție, menținerea nivelurilor optime a stocurilor și asigurarea unei utilizări eficiente a resurselor. De asemenea, managementul producției se concentrează pe optimizarea fluxurilor de lucru și pe îmbunătățirea continuă a proceselor pentru a crește productivitatea și reduce costurile. [2]

Un alt element cheie este calitatea (controlul calității). Managementul producției nu se limitează doar la cantitate, ci și la asigurarea standardelor de calitate înalte. Acest lucru este realizat prin implementarea unor sisteme riguroase de control al calității și prin adoptarea unor practici precum managementul total al calității (TQM) și al principiilor Six Sigma. [3]

În concluzie, managementul producției reprezintă un domeniu dinamic și complex, care necesită o înțelegere profundă a proceselor de producție, tehnologiilor emergente și principiilor de management eficient.

1.1.1. Planificarea producției

În timp ce producția planificată se concentrează în special în domeniul producției de serie, în această secțiune, planificarea producției este abordată sub aspect critic pentru a studia eficiența și eficacitatea acesteia. În acest sens, această secțiune explorează metodele de planificare a producției, discută direct modul în care acestea sunt legate de calitate și costurile produsului finit. [4] Prin urmare, planificarea producției implică elaborarea de obiective și termene clare și luarea unor decizii în ceea ce privește nevoile și resursele companiei. Aceasta (planificarea) include estimări privind: cererea, cantitatea de produs necesară, controlul resurselor și programarea. [5]

Prin urmare, producția de serie plasează un proces de planificare al producției concentrat pe fluxurile de producție pentru a minimiza timpul necesar și a maximiza eficacitatea alocării resurselor. [6] O chestiune elementară a planificării producției de serie este echilibrarea între cerere și ofertă. Pentru a realiza aceasta, este necesar a fi avută o cunoaștere amănunțită a contextului dinamic al competiției și al operaționalizării resurselor. Metodele moderne, cum ar fi planificarea și programarea agregată, sunt utilizate din ce în ce mai des. [7]

Tehnologia joacă un rol semnificativ în modernizarea planificării producției. Sistemele informatice avansate, precum planificarea resurselor întreprinderii (ERP) și sistemele de execuție a producției (MES), permit managerilor să analizeze datele în timp real și să ia decizii pentru a îmbunătăți procesele de producție. [2]

Derivând din funcțiile generale ale managementul (planificarea, organizare, conducere, control) și luând în considere literatura de specialitate în ceea ce privește managementul modern al producției, putem formula următoarele funcții aplicate în ceea ce privește managementul producției:

I. Planificarea și programarea producției - această funcție combină planificarea strategică cu programarea detaliată a producției. Planificarea își propune stabilirea obiectivelor pe termen lung și scurt, determinarea volumului de producție necesar și configurarea fluxurilor de producție, pentru a maximiza eficiența și a minimiza costurile. Programarea este o activitate mai detaliată care presupune alocarea resurselor, stabilirea duratei de începere și terminare a diferitelor operații și gestionarea priorităților de fabricație. Utilizarea tehnologiilor moderne, cum ar fi software-ul ERP și sistemele avansate de programare, contribuie semnificativ la îmbunătățirea acestei funcții. [8]

II. Flexibilizarea și automatizarea producției - flexibilizarea designului proceselor de producție permite îmbunătățirea schimbărilor cererii sau ale pieței. Automatizarea acționează pentru a optimiza operațiunile repetitive, utilizând roboții și inteligența artificială și crește precizia și consistența output-ului. Această funcție este crucială pentru competitivitate într-un mediu economic atât de dinamic. [9]

III. Controlul și asigurarea calității - controlul producției include monitorizarea continuă a proceselor de producție și reacționarea rapidă la schimbările de la cerințe. Asigurarea calității se orientează către menținerea calității produselor la nivelurile dorite și include inspecții regulate, testare și audit. Controlul statistic al proceselor și utilizarea feedback-ului pentru îmbunătățire continuă a procesului este vital pentru managementul producției. [10]

IV. Sustenabilitatea în producție - acest aspect se concentrează pe impactul cu mediul al proceselor de producție și pe consumul responsabil al resurselor naturale. Acest lucru satisface minimizarea consumului de energie și materiale, reducerea deșeurilor prin reciclare și reutilizare și respectarea standardelor de mediu. Sustenabilitatea devine din ce în ce mai mult un factor de decizie competitiv. [11]

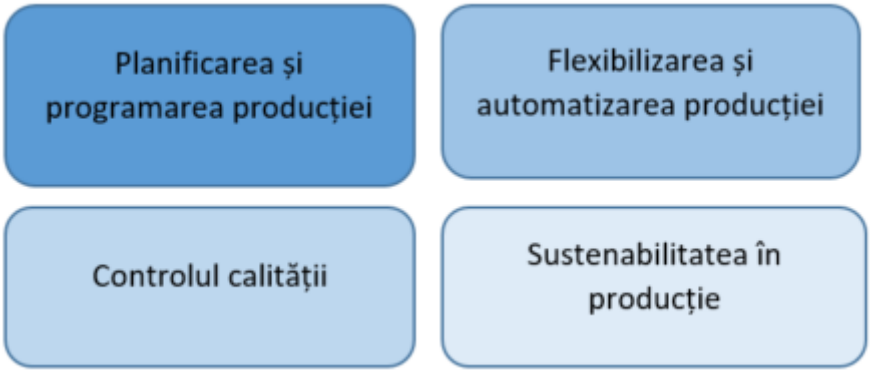


Figura 1.1. Etapele managementului producție
(sursa: prelucrare proprie)

În concluzie, planificarea producției în contextul managementului producției de serie este o sarcină complexă care necesită o abordare strategică și detaliată. Prin echilibrarea cererii cu capacitatea de producție și utilizarea tehnologiilor avansate pentru optimizarea proceselor, organizațiile pot atinge un nivel înalt de eficiență și competitivitate pe piață.

1.1.2. Programarea producției

Prin programarea producției se înțelege repartizarea în timp și spațiu a sarcinilor de producție pe o perioadă de timp considerată, respectând structura produselor și tehnologiile de fabricație ale acestora, în condițiile folosirii raționale a resurselor și a respectării condițiilor contractuale. Ea vizează toate activitățile din sistemul de producție: fabricație, aprovizionare, desfacere, transport etc. Dintre toate acestea programarea fabricației deține ponderea cea mai mare. [12]

În această subsecțiune se explorează procesul de programare a producției în cadrul managementului producției de serie. Mai exact, se examinează cum organizarea și coordonarea eficientă a activităților de producție sunt vitale pentru asigurarea unei fabricații eficiente, punctuale și conforme cu standardele de calitate. Astfel:

- se subliniază importanța programării producției pentru optimizarea utilizării resurselor, reducerea timpilor de așteptare, minimizarea nivelurilor de stoc și asigurarea unei livrări la timp a produselor. În producția de serie, unde se lucrează cu volume mari și cerințe repetitive, o programare precisă și flexibilă este esențială;
- tehnologia informației și automatizarea joacă un rol cheie în facilitarea programării producției. Sistemele informatice avansate, cum ar fi planificarea resurselor întreprinderii (ERP) și sistemele de planificare și programare avansate (APS), sunt esențiale pentru analiza datelor în timp real și luarea de decizii informate; [13]
- Prin procesul de programare există o serie de provocări cum ar fi: variabilitatea cererii, constrângerile de capacitate și schimbările neașteptate în cerințe sau aprovizionare. Aceasta implică găsirea echilibrului între flexibilitate și eficiență; [14]

Programarea producției reprezintă un element esențial în managementul modern al fabricării, având scopul de a coordona eficient toate activitățile implicate, astfel încât să se atingă un nivel optim de productivitate și eficiență. În plus față de organizarea sarcinilor de producție, aceasta contribuie la prevenirea riscurilor asociate întârzierilor sau ineficiențelor prin alocarea corectă a resurselor și monitorizarea constantă a proceselor. [15]

1.1.3. Diversitatea tipurilor de producție

În aceast subcapitol se analizează diferitele abordări și metode de producție utilizate în industrie, subliniind modul în care acestea se potrivesc cu cerințele specifice și cu contextul producției de serie. Diversitatea acestor tipuri de producție reflectă adaptabilitatea și flexibilitatea necesară în domeniul industrial pentru a răspunde la variații în cererea de piață, tehnologie și strategii de afaceri.

a) Producția în masă. Acest tip de producție este caracterizat prin fabricarea unor volume mari de produse standardizate. În producția în masă, procesele sunt optimizate pentru eficiență și scădere a costurilor unitare, fiind adesea automatizate pentru a atinge o producție de volum mare. Acest model este eficient pentru produse cu cerere mare și constantă pe piață, dar poate fi mai puțin flexibil în cazul schimbărilor rapide de cerere sau a personalizării produsului. [16]

b) Producția în loturi (Batch Production). Producția în loturi implică fabricarea unui număr limitat de produse simultan. Aceasta permite mai multă flexibilitate și adaptabilitate față de producția în masă, permițând producerea mai multor variante de produse și adaptarea la schimbările cererii de pe piață. Producția în loturi este ideală pentru produse care necesită mai multe variante sau personalizări. [17]

c) Producția la comandă (Make-to-Order). În acest model, producția începe numai după primirea unei comenzi specifice de la client. Acest tip de producție permite un grad ridicat de personalizare a produsului și este adesea utilizat în industrii unde produsele trebuie să fie adaptate cerințelor unice ale clienților. Eficiența și gestionarea eficientă a timpului de producție sunt esențiale în acest model. [18]

d) Producția „Lean". Inspirată din Sistemul Toyota de producție, producția „lean" se concentrează pe eliminarea risipei și îmbunătățirea continuă. Prin analizarea și optimizarea fiecărui aspect al procesului de producție, producția „lean" urmărește să maximizeze eficiența și să reducă costurile și timpul de producție. [19]

e) Producția flexibilă. Acest tip de producție folosește echipamente și procese care pot fi rapid adaptate pentru a produce o varietate de produse. Este ideală pentru piețele cu cerințe în schimbare rapidă și pentru companiile care doresc să mențină o capacitate de răspuns rapid la cerințele diverse ale clienților. [20]

f) Producția Just-in-Time (JIT). Acest model se bazează pe producerea produselor numai atunci când sunt necesare, reducând astfel stocurile și costurile de stocare. Producția JIT necesită o coordonare strânsă cu furnizorii și o planificare precisă a producției.

Prin „*tip de producție*" se înțelege o stare organizatorică şi funcțională a întreprinderii, determinată de nomenclatura produselor fabricate, volumul producției executate pe fiecare poziţie din nomenclatură, gradul de specializare a întreprinderii, secţiilor şi locurilor de muncă, modul de deplasare a diferitelor materii prime, materiale, semifabricate de la un loc de muncă la altul.

În practică se disting 3 tipuri de producţie [21]:

- tipul de producţie în serie;
- tipul de producţie în masă;
- tipul de producţie individual.

Practica arată însă, că în cadrul întreprinderilor de producţie industrială nu există un tip sau altul de producţie în formele prezentate, ci în cele mai multe cazuri pot să coexiste elemente comune din cele trei tipuri de producţie. În acest caz, metoda de organizare a producţiei va fi adecvată tipului

de producţie care are cea mai mare pondere în întreprindere, precum şi în funcţie de condiţiile concrete existente.

Tipul de producţie în serie

Tipul de producţie în serie este şi el de mai multe tipuri, în funcţie de mărimea lotului de fabricaţie, şi anume:

- tipul de producţie de serie mare;
- tipul de producţie de serie mijlocie;
- tipul de producţie de serie mică. [22]

1.2. Inovații și tendințe actuale în planificarea producției

1.2.1. Impactul tehnologiei IoT

Internetul Lucrurilor (IoT) a revoluționat multe sectoare, inclusiv producția industrială, oferind noi oportunități pentru eficiență și adaptabilitate. Integrarea tehnologiei IoT în procesele de producție permite companiilor să monitorizeze în timp real echipamentele și procesele de producție, folosind senzori conectați care colectează date despre starea mașinilor, performanța producției și condițiile de lucru. Aceste informații permit managerilor să identifice rapid problemele și să intervină pentru a optimiza producția, ducând astfel la o îmbunătățire semnificativă a calității produselor și a eficienței operaționale. [23]

Tehnologia IoT facilitează automatizarea proceselor de producție. Mașinile inteligente, dotate cu capacități de învățare automată și analiza datelor, pot ajusta automat parametrii de producție pentru a maximiza eficiența și a reduce risipa. De exemplu, un sistem IoT poate ajusta automat viteza unei linii de asamblare în funcție de cerințele curente de producție. [24]

Unul dintre cele mai valoroase avantaje ale IoT în producție este întreținerea predictivă. Analiza datelor colectate de la senzori permite anticiparea defectelor înainte de a se produce, permițând intervenții preventive care reduc timpul de inactivitate și costurile de reparații. Zonta et al. (2020) subliniază că întreținerea predictivă poate reduce semnificativ costurile și poate îmbunătăți fiabilitatea echipamentelor. [25]

IoT oferă posibilitatea de a personaliza producția în masă. În producția de serie, IoT poate permite schimbări rapide ale setărilor pentru a produce variații ale unui produs, răspunzând astfel la cerințele specifice ale clienților. Conectivitatea și sincronizarea datelor între furnizori, producători și distribuitori îmbunătățesc lanțurile de aprovizionare, oferind o imagine clară a întregului proces de producție, de la materii prime la produsul finit, și optimizând gestionarea stocului și livrarea. [26]

Pe baza studiilor asupra practicilor din domeniu, se identifică cinci jucători-cheie ai ecosistemului IoT [27]:

1. Dezvoltatorii de platforme software (de exemplu, dezvoltatori de platforme de date, dezvoltatori de platforme de securitate, furnizori de servicii cloud).
2. Dezvoltatorii de platforme hardware (de exemplu, producători de plăci, dezvoltatori de controlere, producători de dispozitive gateway, producători de senzori).
3. Dezvoltatorii de tehnologii de rețea (de exemplu, companii de telecomunicații, dezvoltatori de platforme de conectivitate).

4. Dezvoltatorii de aplicații/soluții (de exemplu, dezvoltatori de analize de date, dezvoltatori de aplicații, integratori de sistem).
5. Utilizatori și clienți (de exemplu, utilizatori corporativi, clienți corporativi, clienți individuali).

Acești actori colaborează pentru a crea un ecosistem IoT robust și eficient, care poate transforma radical industria producției prin îmbunătățirea eficienței, reducerea costurilor și creșterea flexibilității.

Revoluția IoT în producția industrială reprezintă un pas semnificativ spre o industrie mai eficientă și adaptabilă. Prin monitorizarea în timp real, automatizarea proceselor și întreținerea predictivă, IoT transformă modul în care companiile operează, oferind avantaje competitive importante și deschizând noi oportunități de personalizare și optimizare a producției. Această transformare nu este posibilă fără contribuția concertată a tuturor actorilor din ecosistemul IoT, de la dezvoltatori de tehnologii până la utilizatori finali. [28]

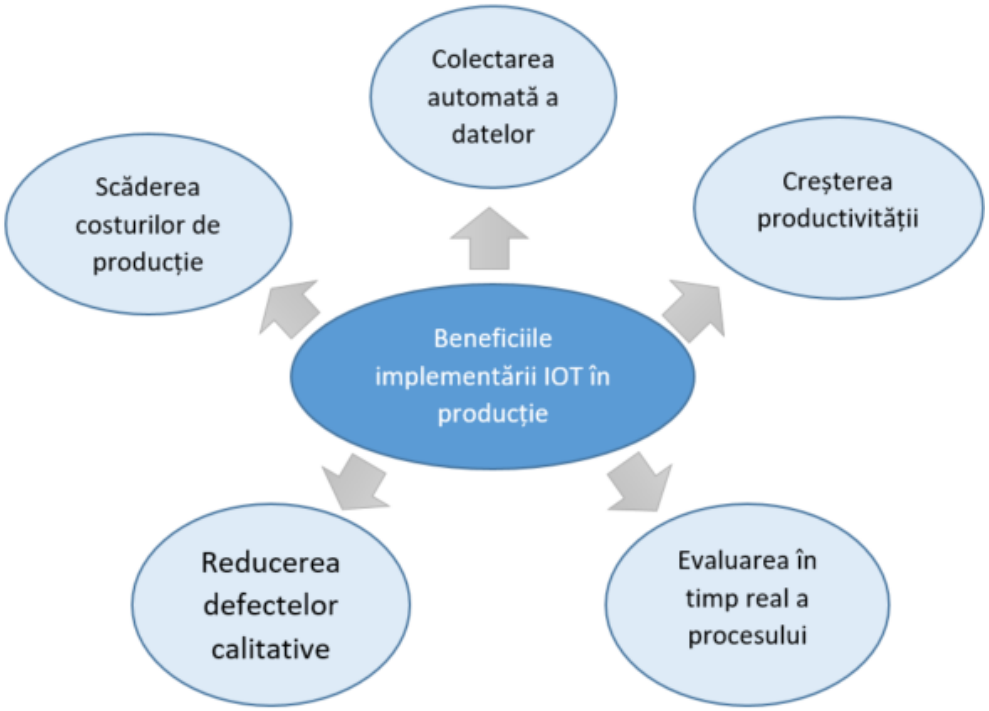


Figura 1.2. Beneficiile implementării IoT în producție
(sursa: prelucrare proprie)

1.2.2. Abordarea problemei Job Shop Schedule (JSS)

Problema Job Shop Schedule se referă la alocarea optimă a unui set de joburi (sau sarcini) pe un număr de mașini sau resurse, unde fiecare job are un anumit traseu prin atelier și necesită anumite operațiuni care trebuie executate într-o ordine specifică. Scopul este de a maximiza eficiența producției, minimizând timpul total necesar pentru finalizarea tuturor joburilor sau timpul de așteptare al mașinilor. [29]

În producția de serie, **problema JSS devine și mai complexă datorită** volumelor mari de producție și diversității operațiunilor. Planificarea eficientă a producției în acest context necesită o analiză detaliată a secvențelor de operațiuni, a capacității echipamentelor și a priorităților de producție.

Metode de soluționare a problemei JSS:

- **algoritmi de optimizare**: utilizarea algoritmilor de optimizare, cum ar fi algoritmii genetici, programarea liniară sau algoritmii evolutivi, pentru a găsi soluții eficiente la problema JSS. [30]
- **sisteme bazate pe IA și machine learning**: implementarea sistemelor de inteligență artificială și învățare automată care pot analiza datele de producție și pot genera soluții de programare optimizate.
- **simulări și modelare**: folosirea simulărilor pentru a modela diferite scenarii de programare și a evalua impactul acestora asupra eficienței producției.

Impactul soluționării eficace a JSS în producția de serie:

- **reducerea ciclurilor de producție**: optimizarea programării producției contribuie la reducerea timpilor de așteptare și a timpilor morți, crescând astfel capacitatea de producție. [31]
- **flexibilitate și adaptabilitate**: o soluție eficientă la problema jss permite flexibilitatea în ajustarea planurilor de producție în funcție de schimbările cererii sau de alte condiții variabile.
- **îmbunătățirea satisfacției clientului**: prin optimizarea programării, se poate răspunde mai rapid cerințelor clienților, îmbunătățind astfel nivelul de satisfacție a acestora.

1.3. Metode contemporane de planificare a producției în serie

Industria modernă se confruntă cu provocări complexe, cum ar fi fluctuații ale cererii, presiuni competitive, inovații tehnologice rapide și cerințe tot mai mari de personalizare a produselor. În acest mediu, planificarea tradițională a producției, care se bazează pe modele și metode rigide, nu mai este suficientă pentru a răspunde eficient acestor provocări. [32]

Avantajele metodelor contemporane [33]:

- **flexibilitate și adaptabilitate**: metodele moderne de planificare permit o mai mare flexibilitate și adaptabilitate, esențiale pentru a răspunde rapid schimbărilor de pe piață. Ele permit ajustarea producției în funcție de cererea fluctuantă, minimizând astfel risipa și optimizând utilizarea resurselor.
- **integrarea tehnologiilor avansate**: metodele contemporane integrează tehnologii avansate, cum ar fi IOT, IA și analiza de date, care permit o mai bună înțelegere și anticipare a tendințelor pieței și a comportamentului consumatorilor.
- **optimizarea eficienței și productivității**: prin folosirea unor metode de planificare mai sofisticate, cum ar fi programarea bazată pe constrângeri sau algoritmi de optimizare, se pot identifica cele mai eficiente moduri de a aloca resursele și de a programa operațiunile, crescând astfel productivitatea.

Impactul asupra producției de serie [34] **:**

- **capacitatea de a răspunde cererii personalizate:** metodele moderne permit producătorilor să ofere produse personalizate fără a sacrifica eficiența producției în masă.
- **reducerea costurilor și creșterea calității**: prin optimizarea proceselor, aceste metode pot reduce costurile și pot îmbunătăți calitatea produselor.
- **competitivitate îmbunătățită:** adoptarea metodelor moderne de planificare poziționează companiile în avangarda competitivității pe piața globală.

În contextul tezei, acest capitol subliniază importanța adoptării metodelor moderne și tehnologiilor avansate pentru a face față provocărilor dinamice ale pieței și pentru a îmbunătăți continuu eficiența și flexibilitatea în producția de serie. Prin integrarea acestor metode contemporane, producătorii pot atinge un nivel superior de performanță operațională, pot răspunde mai eficient cerințelor pieței și pot îmbunătăți sustenabilitatea proceselor de producție.

1.3.1. Funcția obiectiv - Generalități

Generalul D. Eisenhower afirma că „planurile sunt doar hârtie, dar planificarea este esențială”. Planificarea reprezintă un proces continuu, în care programele deja existente se combină cu prognoze noi, estimări și comenzi noi din portofoliu.

Programarea producției este considerată cea mai complexă funcție a procesului de producție. Prin această funcție se stabilește modul în care realizarea unui produs este implementată în procesul de fabricație. Se aleg tehnologia și itinerariul tehnologic, iar definirea procesului de producție se face în funcție de caracteristicile piesei de prelucrat (forme, dimensiuni, material, toleranță, rugozitate, funcționalitate, număr de piese etc.). [35]

Termenul „funcție obiectiv” a fost introdus de matematicianul George Dantzig (1914-2005) în domeniul planificării și programării, în special al programării liniare. Dantzig a creat modele matematice pentru a implementa decizii la scară largă, marcând începutul unei noi ere în procesul de luare a deciziilor. Funcția obiectiv este o expresie matematică numerică ce reflectă obiectivul care trebuie atins de program.

O funcție obiectiv evaluează dacă un vector este fezabil, adică dacă coordonatele acestuia îndeplinesc toate condițiile secundare impuse. Deși termenul a fost introdus de Dantzig, conceptul a existat cu mult timp înainte, un exemplu fiind metoda multiplicatorilor, dezvoltată de Joseph-Louis Lagrange (1736-1813) pentru probleme de optimizare cu constrângeri de egalitate. Acest concept a avut diverse denumiri, fiind aplicat în diferite probleme de optimizare, indiferent de domeniu. [36]

Din punct de vedere statistic, s-a conturat ideea esențială că alegerea unei funcții de estimare, în conformitate cu variația reală observată într-un anumit context aplicat, și selecția metodei statistice pentru modelarea unei probleme sunt dependente de înțelegerea costurilor generate de circumstanțele specifice problemei. [37]

Astfel, alegerea funcției obiectiv este determinată în mod direct de performanța algoritmilor, care sunt concepuți pentru a obține cele mai bune valori posibile pentru funcție, ajustând parametrii sistemului în consecință. Problema de optimizare reprezintă o aplicație matematică prin care se identifică o soluție optimă dintr-o multitudine de variante, evaluând funcția obiectiv și respectând constrângerile impuse. Multe probleme din matematică, statistică, inginerie, economie și științele aplicate pot fi formulate ca probleme de optimizare. [38]

1. Începând cu anul 1984, în cadrul UTCN, a fost definit conceptul de planificare agregată a producției. [39] Acesta descrie caracteristicile unui sistem de producție cu stadii multiple, sistemul ierarhic de planificare a producției și modul de aplicare a programării liniare. De asemenea, sunt identificate funcțiile obiectiv care stau la baza programării producției, ținând cont de stocuri, costuri de producție, capacitate de producție și ore de muncă:

$$Min\ Z = \sum_{i=1}^{I} * \sum_{t=1}^{T} * (c_{it} * X_{it} + s_{it} * S_{it)} + \sum_{t=1}^{T} o_t * O_t \quad (1)$$

cu următoarele condiții:

$$S_{i,t-1} + X_{it} - S_{it} = l_{it} \quad \text{i=1…I ; t=1…T,} \quad (2)$$

$$\sum_{i=1}^{I} * m_i * X_{it} \leq r_t + O_t \quad \text{t=1…T,} \quad (3)$$

$$O_t \leq (adm)_t \quad \text{t=1…T,} \quad (4)$$

$$X_{it} \geq 0 \quad \text{i=1…I ; t=1…T,} \quad (5)$$

$$S_{it} \geq 0 \quad \text{i=1…I ; t=1…T,} \quad (6)$$

$$O_t \geq 0 \quad \text{t= 1…T,} \quad (7)$$

Unde:

i = grupa de produse i=1…I

t = perioada de timp t=1…T

X_{it}= cantitatea din grupa i produsă în perioada t

S_{it}= cantitatea din grupa i aflată în stoc la sfârșitul perioadei t

r_t= capacitatea de producție (ore) în perioada t

O_t= capacitatea suplimentară (ore) necesară în perioada t

c_{it}= costul producției articolelor i în perioada t

s_{it}= cost stocare a articolelor **i** în perioada t

o_t= costul unei ore suplimentare de capacitate în plus față de ora de program reglementat

l_{it}= cantitatea prevăzută a se livra de articole i în perioada t

m_i= consum normat de capacitate de producție pe unitatea de produs (inversul productivității)

$(adm)_t$= numărul maxim de ore de capacitate suplimentară admis de lege

X, S si O – reprezintă necunoscutele modelului

„Min Z” reprezintă minimizarea funcției Z, care este funcția obiectiv, constând în totalitatea costurilor implicate de fabricarea grupelor de produse „I” pe durata orizontului de planificare, format din perioadele „T”. Prin convenție, se admite că toate livrările au loc la sfârșitul perioadei „t”. Setul de constrângeri (2) reprezintă ecuațiile de „conservare” a fluxurilor materiale, indicând că livrările din perioada t pot proveni din două surse: stocul rămas din acea perioadă (cost stocare a articolelor **i** în perioada **t**) și producția curentă (cantitatea din grupa **i** produsă în perioada **t).**

Din condiția (2) și din cele două condiții de nenegativitate (5) și (6), rezultă că este necesară respectarea planului de livrări în fiecare perioadă. Pentru t=1, S_{i0} reprezintă stocul inițial din grupa i, existent la începutul primei perioade a orizontului de planificare. Constrângerile (3) definesc, în

partea stângă, necesarul de capacitate în fiecare perioadă t, care nu poate depăși disponibilul provenit din ore normale și ore suplimentare.

Din natura modelului, rezultă că toate stocurile S_{it} la sfârșitul orizontului de planificare vor avea valoarea zero. Dacă acest lucru trebuie evitat, modelului i se va impune să nu planifice valorile stocurilor sub o valoare considerată adecvată.

2. Studiile unui grup de cercetători de la Department of Business Administration, National Central University Taouyan Taiwan [40] [41] [42] , au avut ca bază stabilirea funcției obiectiv în cazul unui producător de anvelope ce are un proces de producție cu reducere de emisii de carbon, pentru maximizarea profitului:

Maxim π= (Venituri totale) – (Costuri directe cu material) – (Costuri directe cu forța de muncă) - (Costuri de manipulare) – (Taxa de poluare cu carbon) – (Alte costuri fixe)

Adică:

$$\text{Maximize } \pi = \sum_{i=1}^{n} S_i Q_i \; - \sum_{k=1}^{s}\{(C_k \sum_{i=1}^{n}(\alpha_{ik} Q_i))\} \; - \; -\{[(LC_0 + \phi_1(LC_1 - LC_0) + \phi_2(LC_2 - LC_0))]\} - \sum_{i=1}^{n} u_{ij} b_i \; - (CEC_1\delta_1 + CEC_2\delta_2 + CEC_3\delta_3) - F \qquad (1)$$

Unde avem:

Variabile:

Q_i cantitatea din produsul i produsă de compania T

$ø_1$, $ø_2$ – setul de variabile 0-1 din tipul 1 a setului special de comenzi, când doar una din variabile e diferită de zero

ε_1, ε_2, ε_3 – setul de variabile 0-1 din tipul 1 a setului special de comenzi, când doar una din variabile e diferită de zero

δ_0, δ_1, δ_2, δ_3 – un set de variabile diferite de zero din setul special de comenzi de tip 2 când cel puțin 2 variabile adiacente sunt diferite de zero

φ_0 , φ_1 , φ_3 – un set de variabile diferite de zero din setul special de comenzi de tip 2 când cel puțin 2 variabile adiacente sunt diferite de zero

b_i – numărul de loturi manipulate din produsul i

Parametri:

S_i – prețul de vânzare pe unitate a produsului i

C_k – costul pe unitate a produsului i

α_{ik} – necesarul de material k pentru a produce o unitate din produsul i

W_k – cantitatea de material k disponibil.

LH_0 – numărul de ore de manoperă directă necesare.

LH_1 – primul nivel de ore maxime suplimentare de muncă

LH_2 – al noilea nivel de ore maxime suplimentare de muncă

LC_0 – costul orelor de muncă normale

LC_1 – costul orelor de muncă suplimentare din primul nivel

LC_2 – costul orelor de muncă suplimentare aferente celui de al doilea nivel

e_j – costurile operaționale ale activităților j

a_{ij} – numărul de ore masină necesare pentru a produce o unitate de produs i în activitatile j=1-6

u_{ij} –numărul de ore mașină necesare pentru a transporta un lot de produse I aferente activității j-7

n_{ij} – cantitatea de produse I ce intră în componența unui lot de activități j.

l_i – necesarul de ore de muncă pentru a produce o unitate din produsul i

TDL – numărul de ore de muncă directă folosite în ecuație

MH_j – numărul de ore mașina disponibile pentru fiecare activitate j

CEC_1 – limita maximă de emisii de carbon din primul nivel de taxare

CEC_2 – limita maximă de emisii de carbon din al doilea nivel de taxare

CEC_3 – limita maximă de emisii de carbon din al treilea nivel de taxare

TCEQ – cantitatea totală de emisii de carbon eliminate în timpul procesului de producție

CE_1 – numărul maxim de emisii de carbon eliminate conform primului nivel de taxare

CE_2 – numărul maxim de emisii de carbon eliminate conform celui de al doilea nivel de taxare

CE_3 – numărul maxim de emisii de carbon eliminate conform celui de al treilea nivel de taxare

q_i – costul cu manevrarea unui lot de produse i

v_i – cantitate de emisii de carbon emise în timpul producției unei unități din produsul i

F – costuri fixe.

3. În anul 2017, în cadrul studiului efectuat cu privire la un algoritm genetic îmbunătățit pentru optimizarea multidimensională a planificării și programării producției [43] [44] [45], Son Duy Dao, Kazem Abhary și Romeo Marian au arătat că scopul principal este acela de a dezvolta în continuare un AG (Algoritm Genetic) mai robust pentru problema de planificare a producției cu constrângeri de precedență cu mai multe linii de producție, într-o variantă îmbunătățită față de încercările lor anterioare. [46]

În acest sens, au arătat că formularea cu numere întregi mixte pentru această problemă este dezvoltată după cum urmează:

Ipoteze

• Compania poate produce orice amestec de diferite produse.

• Compania poate lucra 24 de ore pe zi, 7 zile pe săptămână.

• Procedurile din vânzarea produselor vor fi doar accesibile pentru perioada următoare, așa că nu ar trebui luate în considerare pentru orizontul de planificare curent - perioada de timp curentă.

Indici

i = Indicele liniei de producție

j = Indicele secvenței de producție

k = Indicele produsului

Parametrii

a = Coeficientul de pondere

A = Numărul de puncte ale indicelui de satisfacție a clienților pierdute în fiecare zi de întârziere când fiecare produs este fabricat după termenul limită

B = Numărul de puncte de indice de satisfacție a clienților obținute atunci când fiecare produs este realizat înainte de termenul limită

C = Totalul capitalului de lucru disponibil ($)

D_k = Termenul limită al produsului k (ziua lunii)

E_{ki} = Cheltuiala de producție a produsului k în linia de producție i ($)

M = Total material disponibil (kg)

MR_k = Necesarul de material pentru produsul k (kg)

l = Număr a diferitelor linii de producție

L = Total forță de muncă disponibilă (ore)

LR_{ki} = Necesarul de forță de muncă pentru produsul k în linia de producție i (ore)

O_i = Alte cheltuieli generale pentru funcționarea liniei de producție i ($/oră)

p = Numărul de diferite tipuri de produse disponibile

PC_i = Schimbarea produsului în linia de producție i (ore)

U_k = Penalizare din cauza livrării cu întârziere a produsului k (procent din prețul produsului pe zi de întârziere)

V = Numărul minim de tipuri de produse necesare

R_k = Numărul maxim de zile de întârziere acceptate cu penalizare pentru produsul k; în caz contrar, clientul nu acceptă produsul și acesta va fi returnat companiei

S_k = Pretul de vanzare al produsului k ($)

Variabile de decizie

C_i = Capital de lucru alocat liniei de producție i ($)

M_i = Material alocat liniei de producție i (kg)

L_i = Forță de muncă alocată liniei de producție i (ore)

P_{kij} = Produsul k selectat dintre cele disponibile și realizat în linia de producție *i* în succesiunea lui *j*

Q_k = Cantitatea de produs k

Q_{ki} = Cantitatea de produs k realizată în linia de producție *i*

Hkij = 1 Dacă produsul k este selectat pentru a fi realizat în linia de producție i în secvența j
0 În caz contrar

θ_i = 1 Dacă produsele din secvența j și din secvența j - 1 din linia de producție i nu sunt aceleași
0 În caz contrar

Semnul 1{x} = x Dacă $x < 0$
0 Dacă $x \leq 0$

Semnul 2{x} = 1 Dacă $x > 0$
0 Dacă $x \leq 0$

Unic (x) = O funcție care este capabilă să determine numărul de elemente unice din matricea x.

Funcția obiectiv este aplicată pentru a lua în considerare atât profitul companiei, cât și indicele de satisfacție a clienților. Funcția obiectiv care trebuie maximizată este suma profitului total al companiei și a indicelui de satisfacție a clienților cu un coeficient de pondere dat, care este calculat prin Ecuația (1) unde:

- F este valoarea de funcție obiectiv;
- TI este venitul total;
- MC este costul total de producție;
- OH este costul general total pentru rularea liniilor de producție selectate;
- CD este costul total asociat cu penalizarea datorată produselor realizate după termenul limită;
- CR este costul total datorat produselor returnate sau neacceptate de către client pentru că sunt prea târziu;
- SI este punctele totale ale indicelui de satisfacție a clienților;
- α este coeficientul de greutate.

$$F=\alpha[TI-MC-OH-CD-CR]+(1-\alpha)SI. \quad (1)$$

Componentele costului din Ecuația (1) se calculează după cum urmează:

Venitul total pe care societatea le realizează din comenzile de produse (TI):

$$TI = \sum_{k=1}^{p} Q_k * S_k \quad (2)$$

Costul total de producție al comenzilor de produse (MC):

$$MC = \sum_{k=1}^{p} \sum_{i=1}^{l} Q_{ki} * E_{ki} \quad (3)$$

Costul general total pentru rularea liniilor de producție selectate (OH):

$$OH = \sum_{k=1}^{p} \sum_{i=1}^{l} \sum_{j=l}^{\infty} (\theta_i * PC_i + LR_{ki}) H_{kij} O_i \quad (4)$$

Costul total asociat cu penalizarea datorată produselor realizate după termenul limită (CD):

$$CD = \sum_{k=1}^{p} \sum_{i=l}^{l} \sum_{j=l}^{\infty} \left[\frac{\text{Semnul } 1\{(\theta_i * PC_i + LR_{ki}) H_{kij} - 24D_k}{24} \right] U_k * S_k \quad (5)$$

Costul total datorat produselor returnate sau neacceptate de client pentru că sunt livrate cu întârziere (CR):

$$CR = \sum_{k=1}^{p} \sum_{i-i=1}^{l} \sum_{j=1}^{\infty} [\text{Semnul } 2\{(\theta_i * PC_i + LR_{ki}) H_{kij} - 24D_k - 24R_k\}] S_k \quad (6)$$

Punctele totale ale indicelui de satisfacție a clienților (SI):

$$SI = \sum_{k=1}^{p} \sum_{i=1}^{l} \sum_{j=1}^{\infty} \left[B - \left(\frac{\text{Semnul } 1\{(\theta_i * PC_i + LR_{ki}) H_{kij} - 24D_k\}}{24} \right) A \right] \quad (7)$$

Subiect la:

Cantitatea de produs k:

$$\sum_{i=1}^{l} Q_{ki} = Q_k \quad (8)$$

Capitalul de lucru total disponibil:

$$\sum_{i-1}^{l} C_i = C \quad (9)$$

Materialul total disponibil:

$$\sum_{i=1}^{l} M_i = M \tag{10}$$

Manopera totală disponibilă:

$$\sum_{i=1}^{l} L_i = L \tag{11}$$

Materialul total alocat liniei de producție *i*:

$$\sum_{k=1}^{p} Q_{ki} MR_k \leq M_i \tag{12}$$

Forța de muncă totală alocată liniei de producție *i*:

$$\sum_{k=1}^{p} \sum_{j=1}^{\infty} (\theta_I * PC_i + LR_{ki}) H_{kij} \leq L_i \tag{13}$$

Capitalul de lucru total alocat liniei de producție *i*:

$$\sum_{k=1}^{p} Q_{ki} * E_{ki} + \sum_{k=1}^{p} \sum_{j=1}^{\infty} (\theta_i * PC_i + LR_{ki}) H_{kij} O_i \leq C_i \tag{14}$$

Numărul minim de tipuri de produse necesare:

$$p \geq \text{Unic} \left(\sum_{k=1}^{p} \sum_{i=1}^{l} \sum_{j=1}^{\infty} P_{kij}\right) - 1 \geq V. \tag{15}$$

4. Aceeași Son Duy DAO și Romeo MARIAN, în studiul intitulat *"Optimizarea secvențierii și programării producției utilizând algoritmi genetici*", [43] au reținut că, deși până în prezent au fost elaborate o serie de metode de planificare a producției, problemele de planificare în viața reală nu au fost încă rezolvate complet, iar soluțiile consacrate sunt departe de a fi perfecte din cauza multiplicității constrângerilor implicate.

S-a arătat că funcția obiectiv a fiecărui cromozom este profitul total al companiei. Profitul total se calculează astfel: TP =I - (CP + CR + CD), unde:

TP – profitul total al firmei;

I – venitul total;

CP – costul total de producere a produselor;

CR – costul total de funcționare a companiei;

CD – costul total asociat cu penalizarea datorată produselor realizate după termenul limită.

Calculul venitului total:

$$I = \sum_1^{40} ai * pi \quad (16)$$

Unde:

pi – prețul de vânzare a produsului Ai;

ai – numărul de produse Ai care urmează să fie produse/vândute;

ai = 0 dacă produsul Ai nu este selectat pentru a produce, i = 1,.. 40;

Calculul costului total de producere a pieselor (CP)

$$CP=\sum_1^{40} ai * ci \quad (17)$$

Unde:

ci – costul realizării produsului Ai;

ai – numărul de produse Ai care urmează să fie produse;

ai = 0 dacă produsul Ai nu este selectat pentru a produce, i = 1,.. 40.

Calculul costului total al companiei de funcționare (CR):

$$CR = 0,17*(h1+h2) \quad (18)$$

Unde:

h1 – ore pentru producerea produselor selectate;

h2 – ore pentru schimbarea produsului;

0,17 – cost pe oră de funcționare a companiei.

Calculul costului penalităților totale datorat produselor fabricate după termenul limită (CD)

$$CD = \sum_1^{40} 0.05 * ti * ai * pi \quad (19)$$

Unde:

pi – prețul de vânzare a produsului Ai;

ti – numărul de zile în care produsul Ai este realizat după termen;

ai – numărul de produse Ai care urmează să fie produse;

ai = 0 dacă produsul Ai nu este selectat pentru a fi produs;

0,05 – penalizare de întârziere (5% din prețul inițial/zi), i = 1,.. 40;

Reiese că forța totală de muncă în ore ale companiei este de 650 pe o lună de 30 de zile. Așadar, în medie, în fiecare zi compania rulează k (ore), k=(650/30) = 21,27.

Având ca suport aceste informații și succesiunea de producere a produselor, *ti* din ecuația de mai sus poate fi determinată după cum urmează

$$ti=[(\mathrm{Si}+\mathrm{Ti})-\mathrm{Di}*\mathrm{k}]/\mathrm{k} \quad (20)$$

Unde:

Di – termenul limită al produsului Ai,

k – timpul mediu zilnic de funcționare al companiei,

Si – ora de începere a producerii produsului Ai;

Ti – timpul de procesare pentru produsul Ai; i = 1,.. 40.

1.3.2. Aplicații practice ale funcției obiectiv în alte sectoare.

Funcția obiectiv reprezintă un element fundamental în matematică și în științele aplicate, având scopul de a optimiza diferite procese prin maximizarea sau minimizarea unei anumite valori. În contextul optimizării, funcția obiectiv este utilizată pentru a evalua performanța diferitelor soluții propuse și pentru a ghida procesul de căutare către soluția optimă. [47] Metoda Guinet, prezentată pe larg în capitolul 3.2, una dintre tehnicile avansate de optimizare, a fost dezvoltată inițial pentru a aborda problemele complexe din programarea producției de serie. Cu toate acestea, eficiența și flexibilitatea sa au permis aplicarea acestei metode într-o varietate de alte domenii, oferind soluții inovatoare pentru optimizarea resurselor și a proceselor.

Utilizată inițial pentru optimizarea programării producției, s-a dovedit a fi un instrument puternic aplicabil și în alte domenii, cum ar fi logistica, agricultura, sănătatea și sectorul energetic. Această funcție se bazează pe tehnici de programare matematică și inteligență artificială, permițând optimizarea resurselor și a proceselor complexe.

Logistica și managementul lanțului de aprovizionare. Un domeniu important în care funcția Guinet poate fi aplicată este logistica și managementul lanțului de aprovizionare. În acest sector, optimizarea rutelor de transport, gestionarea stocului și alocarea resurselor sunt critice pentru

reducerea costurilor și creșterea eficienței. [48] Conform literaturii, utilizarea funcției Guinet și a programării matematice în logistica urbană poate duce la o reducere semnificativă a costurilor operaționale prin optimizarea rutelor și gestionarea dinamică a flotelor de vehicule.

Funcția Guinet poate ajuta la determinarea celor mai eficiente rute de livrare, reducând astfel costurile de combustibil și timpul de livrare. [49] De exemplu, un model de programare liniară poate fi folosit pentru a minimiza costurile de transport și a maximiza utilizarea eficientă a vehiculelor. În plus, aplicarea acestei funcții poate îmbunătăți gestionarea stocurilor, asigurând disponibilitatea produselor atunci când și unde este necesar, fără a menține stocuri excesive.

Agricultura. În agricultură, funcția Guinet poate fi aplicată pentru optimizarea utilizării terenurilor, gestionarea resurselor de apă și planificarea recoltelor. Aceasta poate ajuta fermierii să maximizeze randamentul culturilor și să minimizeze utilizarea resurselor, contribuind astfel la agricultura sustenabilă. Un exemplu de aplicare a funcției Guinet în agricultură este utilizarea unui model de programare liniară pentru a optimiza rotația culturilor, astfel încât să se maximizeze producția și să se minimizeze eroziunea solului. [50]

Literatura de specialitate arată că optimizarea programării poate fi aplicată și în agricultura de precizie pentru a gestiona mai bine irigațiile și fertilizarea. Aceasta poate duce la economii semnificative de apă și nutrienți, îmbunătățind în același timp randamentele culturilor

Sănătatea. În sectorul sănătății, funcția Guinet poate fi utilizată pentru a optimiza programarea personalului medical, gestionarea resurselor spitalicești și planificarea intervențiilor chirurgicale. De exemplu, un model de programare matematică poate ajuta la optimizarea programelor de lucru ale personalului medical, astfel încât să se asigure o acoperire adecvată în toate secțiile unui spital și să se reducă oboseala personalului. [51]

Studiile arată că aplicarea tehnicilor de optimizare în gestionarea resurselor spitalicești poate îmbunătăți semnificativ eficiența și calitatea serviciilor medicale. De exemplu, un model de programare liniară poate fi utilizat pentru a optimiza alocarea sălilor de operație, minimizând timpul de așteptare pentru pacienți și maximizând utilizarea eficientă a echipamentelor chirurgicale. [52]

Sectorul energetic. Funcția Guinet poate juca un rol crucial și în sectorul energetic, unde optimizarea resurselor și a proceselor este esențială pentru asigurarea unei furnizări fiabile și eficiente de energie. În special, tehnicile de programare matematică pot fi folosite pentru a optimiza distribuția energiei, gestionarea rețelelor de transport și stocarea energiei.

Un exemplu notabil este utilizarea funcției Guinet pentru a optimiza operarea rețelelor inteligente de energie *(en. smart grids*). Aceste rețele utilizează senzori și tehnologii de comunicații avansate pentru a monitoriza și gestiona fluxurile de energie în timp real. Un model de programare

liniară poate fi folosit pentru a optimiza distribuția energiei, minimizând pierderile și asigurând furnizarea continuă de energie către consumatori. [53]

Alte domenii. Funcția Guinet poate fi aplicată și în alte domenii, cum ar fi: a) industria trasporturilor -optimizarea rutelor de transport public și privat pentru a reduce costurile și timpul de deplasare; [54] b) proiectarea și planificarea urbană - utilizarea tehnicilor de programare matematică pentru a optimiza utilizarea terenurilor și a infrastructurilor urbane; c) managementul resurselor umane - optimizarea programelor de formare și dezvoltare a angajaților pentru a maximiza productivitatea și satisfacția la locul de muncă.

1.3.3. Contribuția algoritmilor de inteligență artificială

Algoritmii de inteligență artificială oferă instrumente avansate pentru analiza și prelucrarea datelor complexe în producția de serie. Acești algoritmi pot procesa cantități mari de informații pentru a identifica modele, tendințe și potențiale probleme în fluxul de producție. De exemplu, IA poate fi utilizat pentru a anticipa cerințele de producție, a optimiza alocarea resurselor și a îmbunătăți eficiența liniilor de asamblare.

La ce poate fi folosit IA în producție:

- **predicții și analize**: IA poate oferi predicții precise privind cerințele de piață și comportamentul consumatorilor, permițând producătorilor să ajusteze producția în consecință. [55]
- **îmbunătățirea eficienței și reducerea risipei**: prin analiza continuă a datelor de producție, IA ajută la identificarea rapidă a ineficiențelor și a oportunităților de reducere a risipei. [56]
- **programarea optimă a producției**: algoritmii IA pot fi folosiți pentru a crea programe de producție care să maximizeze utilizarea echipamentelor și să minimizeze timpul de așteptare.

Beneficii cheie ale integrării IA în producție [57]**:**

- **flexibilitate și adaptabilitate**: IA permite producătorilor să fie mai receptivi la schimbările de pe piață, adaptând rapid producția la cerințele în schimbare.
- **îmbunătățirea calității produselor**: IA poate ajuta la monitorizarea și controlul calității produselor în timp real, reducând astfel rata de defecte și îmbunătățind satisfacția clientului.

- **luare de decizii bazată pe date**: deciziile de producție bazate pe analize IA sunt adesea mai precise și mai eficiente, contribuind la o gestionare optimă a producției.

Contribuția algoritmilor de inteligență artificială în planificarea producției în serie este semnificativă și multifacetică. IA nu doar că sporește eficiența și adaptabilitatea proceselor de producție, dar oferă și un avantaj competitiv, permițând producătorilor să răspundă rapid și eficient la cerințele dinamice ale pieței. Integrarea IA în sistemele de producție reprezintă un pas esențial către o industrie mai inteligentă și mai receptivă la nevoile viitoare. [58]

1.4. Analiza literaturii de specialitate privind baza de date Web of Science – Evoluția cercetării în managementul producției în perioada 2020-2023

Urmare a subiectului ales pentru cercetare, a fost efectuată o centralizare statistică. Această analiză explorează tendințele și schimbările în cercetarea din domeniul managementului producției în perioada 2020-2023. Bazându-se pe date statistice din baza de date Web of Science, analiza reflectă fluctuațiile în numărul de articole publicate și evidențiază direcțiile emergente și subiectele de actualitate din domeniu. Prin examinarea acestor tendințe, putem identifica schimbările majore în abordările și tehnologiile folosite în managementul producției și anticipa viitoarele direcții de dezvoltare.Această perspectivă este relevantă pentru cercetători, practicieni și factori de decizie implicați în domeniul producției, oferind o bază solidă pentru înțelegerea evoluției subiectelor de interes și a factorilor care influențează cercetarea. Căutarea în baza de date a fost efectuată pentru

cuvinte cheie în limba engleză(keywords), iar pentru anul în curs (2024) au fost trecute datele în tabel doar cu titlul orientativ (a se vedea tabelul următor).

Tabel 1.1. Evoluția cercetării în domeniul – conform bazei de date Clarivate

Nr crt	Cuvinte cheie (Keywords)	An	Articole
1	"Production Programing/Production Schedule"	2024	124
		2023	8261
		2022	10261
		2021	10695
		2020	10026
2	"Mass Production Programing/ Mass Production Schedule"	2024	4
		2023	426
		2022	481
		2021	518
		2020	496
3	"Mass Production Optimisation"	2024	34
		2023	1676
		2022	2277
		2021	1814
		2020	1861
4	"Production Management"	2024	437
		2023	30476
		2022	36013
		2021	36468
		2020	32567
5	"Mass Production Management"	2024	24
		2023	1381
		2022	1534
		2021	1588
		2020	1439
6	"Optimisation Problem"	2024	506
		2023	32979
		2022	42295
		2021	39441
		2020	36420
7	"Objective Functions with constrains"	2024	9
		2023	593
		2022	709
		2021	686
		2020	634
8	"Genetic Algorithms Used for Objective Functions with Constrains"	2024	0
		2023	42
		2022	48
		2021	47
		2020	45
9	"Hybrid Algorithms Used for Objective Functions with Constrains""	2024	0
		2023	23
		2022	31
		2021	19
		2020	30
10	"CPLEX-OPL Used for Objective Functions with Constrains"	2024	0
		2023	2
		2022	3
		2021	4
		2020	1

(sursa: prelucrare proprie conform datelor Clarivate)

Rezultatele analizei acestui tabel ar putea indica următoarele:

1. Programarea producției - În perioada 2020-2023, interesul pentru „Production Programing” și „Production Schedule” a cunoscut o evoluție ascendentă, cu un vârf în 2021, când au fost publicate 10.695 articole. În 2020, numărul de articole publicate era de 10.026, ceea ce reflectă un interes deja solid pentru optimizarea proceselor de producție. Creșterea din 2021 poate fi asociată cu nevoia companiilor de a implementa soluții mai eficiente, în contextul schimbărilor rapide impuse de pandemie și de digitalizarea proceselor. În 2022, acest interes a continuat să crească, atingând 10.261 articole, un semn că domeniul rămâne o prioritate pentru cercetători și practicieni. Totuși, în 2023, numărul articolelor a scăzut la 8.261, marcând o potențială stabilizare, dar păstrând un interes ridicat pentru noi tehnici de programare a producției. Această scădere poate fi legată de integrarea mai largă a soluțiilor automate și de utilizarea algoritmilor avansați, reducând astfel nevoia de cercetări suplimentare.

2. Programarea producției de masă - În ceea ce privește „Mass Production Programing” și „Mass Production Schedule”, perioada 2020-2023 arată o tendință de scădere a interesului. În 2020, au fost publicate 496 de articole, dar acest număr a crescut ușor în 2021, la 518 articole, semnalând încă o preocupare pentru programarea producției de masă. Totuși, din 2022, numărul articolelor a scăzut la 481, iar în 2023 la 426, reflectând o schimbare clară de focus în industrie. Această tendință descendentă sugerează că interesul pentru producția de masă scade pe măsură ce cerințele pieței se orientează mai mult către personalizarea producției și flexibilitate. Producția de masă nu mai este la fel de relevantă în fața noilor modele de afaceri care pun accent pe adaptabilitate și producția în loturi mici, ceea ce explică scăderea continuă a numărului de cercetări.

3. Optimizarea producției de masă - creșterea semnificativă a articolelor despre optimizarea producției de masă în 2021, menținând un nivel constant între 2021 și 2023, indică faptul că acest subiect rămâne relevant pentru industrie. Analiza literaturii de specialitate sugerează că accentul se pune pe găsirea celor mai eficiente metode de producție de masă, folosind tehnici moderne și tehnologii avansate.

4. Managementul producției - această categorie de cercetare a înregistrat o creștere accentuată între 2020 și 2022, urmată de o scădere în 2023. Această tendință reflectă un interes deosebit pentru tehnici inovatoare în managementul producției, urmată de o stabilizare. Analiza literaturii de specialitate indică faptul că perioada de creștere corespunde unei perioade de adaptare a industriei la noi provocări și tehnologii, în timp ce scăderea ar putea fi asociată cu consolidarea metodologiilor.

5. Managementul producției de masă - în ciuda unei creșteri ușoare în 2021, interesul pentru managementul producției de masă s-a stabilizat în următorii ani. Această tendință poate fi atribuită accentului pus pe metode mai flexibile de producție și o trecere de la producția de masă tradițională la producția personalizată. Literatura de specialitate reflectă această schimbare, sugerând că industria încearcă să se adapteze la cerințele consumatorilor pentru produse mai personalizate.

6. Probleme de optimizare – categoria de cercetare a înregistrat o creștere constantă în articole despre problemele de optimizare, cu un vârf în 2022. Această tendință arată că industria continuă să caute modalități eficiente de optimizare a proceselor de producție. Analiza literaturii de specialitate indică faptul că acest interes este generat de necesitatea de a maximiza eficiența și de a minimiza costurile în procesele de producție.

7. Funcții obiectiv cu constrângeri - această categorie a înregistrat o creștere constantă între 2020 și 2023, ceea ce sugerează un interes crescut pentru metodele de optimizare care țin cont de constrângeri. Analiza literaturii de specialitate arată că aceste funcții sunt critice pentru tehnicile moderne de programare și optimizare, fiind utilizate în diverse contexte, de la producție până la logistică și transport.

8. Algoritmi genetici pentru funcții obiectiv cu constrângeri - această categorie de căutare/cercetare a înregistrat o creștere între 2020 și 2022, urmată de o ușoară scădere în 2023, reflectând interesul pentru utilizarea algoritmilor genetici în contextul funcțiilor obiectiv. Analiza literaturii de specialitate arată că, deși algoritmii genetici rămân relevanți, tendința de scădere ar putea indica o preferință pentru alte tehnici de optimizare sau o schimbare în abordările de rezolvare a problemelor.

9. Algoritmi hibrizi pentru funcții obiectiv cu constrângeri - această categorie a înregistrat o creștere între 2020 și 2022, urmată de o scădere în 2023. Interesul pentru algoritmii hibrizi sugerează o tendință de experimentare în abordările de optimizare, dar scăderea ar putea indica faptul că aceste metode hibrid nu au oferit întotdeauna rezultate superioare. Analiza literaturii de specialitate arată că, deși combinarea algoritmilor poate fi eficientă, succesul depinde de contextul specific.

10. Utilizarea CPLEX-OPL pentru funcții obiectiv cu constrângeri: această categorie a înregistrat o tendință descrescătoare între 2020 și 2023, reflectând un interes în scădere pentru utilizarea software-ului specific în rezolvarea problemelor cu constrângeri. Analiza literaturii de specialitate sugerează că scăderea interesului pentru această categorie poate fi asociată cu apariția altor instrumente și tehnologii mai flexibile sau cu o abordare diferită a problemelor de optimizare.

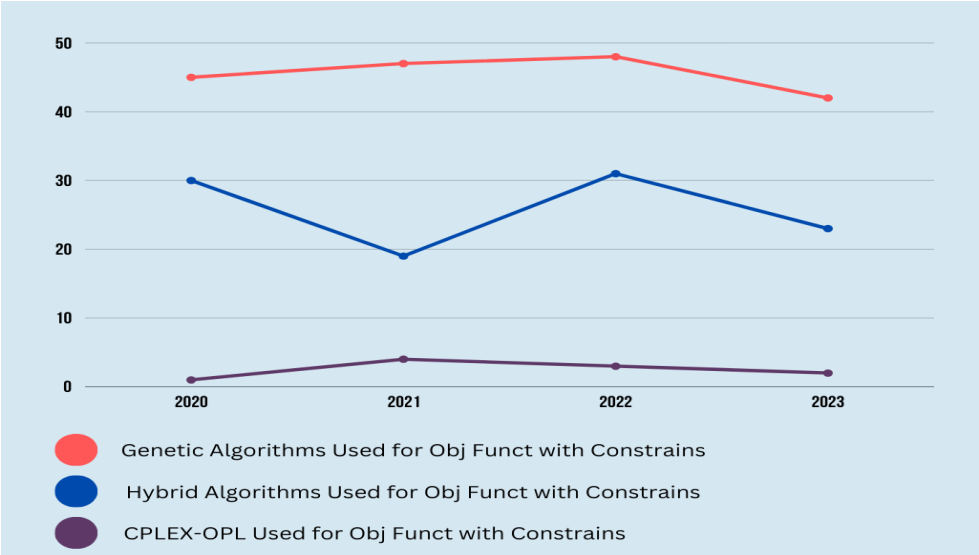


Figura 1.3. Evoluția raportului lunar/anual între utilizarea principalilor algoritmi în literatura de specialitate (sursa: prelucrare proprie)

În concluzie, analiza tendințelor observate în punctele 1-10 indică variații semnificative în interesul față de programarea producției și metodele conexe. Analiza literaturii de specialitate arată că aceste fluctuații reflectă evoluția tehnologiei și schimbările în abordările industriei. Creșterile notabile în anumite domenii, cum ar fi optimizarea programării producției de masă, sugerează perioade de inovație, în timp ce scăderile pot reflecta stabilizarea sau tranziția către abordări mai flexibile.

De asemenea, creșterile și descreșterile în diverse categorii, cum ar fi algoritmii genetici și algoritmii hibrizi, indică o diversificare a metodelor de optimizare utilizate în programarea producției. Literatura de specialitate sugerează că această diversificare reprezintă încercările industriei de a descoperi cele mai eficiente metode de programare și ordonanțare a producției, precum și o tendință de experimentare cu noi tehnici și abordări.

Nu în ultimul rând, din perspectiva impactului tehnologiilor moderne, interesul pentru tehnici precum algoritmii genetici și algoritmii hibrizi indică influența tehnologiilor moderne asupra managementului producției. Analiza literaturii de specialitate subliniază că evoluția tehnologiei și creșterea automatizării au un impact profund asupra metodelor de programare și optimizare. În plus, această tendință sugerează că succesul abordărilor bazate pe tehnologie depinde de capacitatea lor de a se adapta la nevoile și provocările specifice ale industriei.

CAPITOLUL 2: STUDIU COMPARATIV PRIVIND METODELE DE PROGRAMARE A PRODUCȚIEI BAZATE PE ALGORITMI DE INTELIGENȚĂ ARTIFICIALĂ

Capitolul 2 al tezei de doctorat constituie un segment esențial, focalizându-se pe explorarea și compararea diferitelor metode de programare a producției care se bazează pe utilizarea algoritmilor de inteligență artificială (IA). Importanța acestui capitol rezidă în capacitatea sa de a evidenția rolul crucial al IA în transformarea și optimizarea proceselor de producție în serie, un aspect de mare relevanță în contextul industrial actual, unde cerințele de rapiditate, eficiență și flexibilitate sunt în continuă creștere. Prin analiza comparativă a diverselor tehnici bazate pe IA, acest capitol aduce o contribuție semnificativă la înțelegerea modului în care inovațiile tehnologice pot fi aplicate pentru a îmbunătăți programarea și gestionarea producției.

În acest capitol, se urmărește și evaluarea eficacității tehnicilor IA în contextul specific al producției de serie. Se ia în considerare modul în care IA poate influența factori precum timpul de producție, calitatea produselor, costurile operaționale și adaptabilitatea la schimbările pieței. Această analiză comparativă este esențială pentru a identifica cele mai potrivite strategii de IA care pot fi implementate în mediile de producție industriale, contribuind astfel la optimizarea proceselor și la îmbunătățirea productivității.

Capitolul abordează, de asemenea, provocările și oportunitățile aduse de implementarea IA în sectorul producției, oferind o perspectivă realistă asupra beneficiilor și limitărilor acestor tehnologii. Analiza se extinde și asupra aspectelor practice, cum ar fi integrarea soluțiilor de IA în sistemele de producție existente, precum și considerații legate de costuri, formare a personalului și modificări ale infrastructurii

În contextul unei piețe globale din ce în ce mai competitive și a unei nevoi crescânde de eficiență și adaptabilitate, capacitatea de a integra soluții inovatoare de IA în programarea producției reprezintă un avantaj strategic major. Capitolul aduce în prim-plan modalități prin care IA poate îmbunătăți procesele de producție, de la automatizarea deciziilor de planificare până la optimizarea fluxurilor de lucru și reducerea timpilor de inactivitate.

2.1. Abordări ale planificării/programării producției de serie

În contextul producției de serie, eficiența și flexibilitatea sunt esențiale pentru a răspunde rapid și eficient la cerințele pieței și pentru a gestiona resursele în mod optim. Programarea producției este una dintre cele mai complexe funcții ale procesului de producție, implicând stabilirea modului de realizare a produselor prin definirea tehnologiilor și a itinerariilor tehnologice. Aceasta presupune alegerea resurselor și a timpilor de producție pentru o perioadă cuprinsă între 1 și 18 luni, cu obiectivul principal de minimizare a costurilor.

Metodele tradiționale includ planificarea agregată, ordonanțarea producției și organizarea mixtă. Planificarea agregată se concentrează pe determinarea resurselor și timpilor de producție necesari pentru a satisface cererile prevăzute, utilizând tehnici precum programarea liniară pentru a optimiza utilizarea resurselor și pentru a minimiza fluctuațiile forței de muncă și a stocurilor.

Ordonarea producției de serie reprezintă eșalonarea în timp și spațiu a executării operațiilor, având ca scop încărcarea completă a locurilor de muncă și minimizarea duratei ciclului de fabricație. Aceasta include metode de programare continuă, prin priorități și pe bază de stoc, fiecare având avantaje specifice, cum ar fi reducerea așteptărilor și asigurarea continuității producției .

O abordare modernă este utilizarea algoritmilor genetici și a altor tehnici de inteligență artificială pentru optimizarea programării producției. Algoritmii genetici permit generarea unor soluții eficiente pentru probleme complexe, cum ar fi secvențierea și programarea producției, prin simularea proceselor naturale de selecție și evoluție .

Aplicarea modelului Just-in-Time (JIT) și a sistemului Kanban reprezintă alte exemple de metode moderne care îmbunătățesc flexibilitatea și eficiența producției. Metoda JIT se bazează pe sincronizarea proceselor de fabricație pentru a produce exact ceea ce este necesar, atunci când este necesar, reducând astfel stocurile și îmbunătățind fluxul de lucru. Acest subcapitol stabilește o bază solidă pentru discuțiile ulterioare despre cum inovațiile în IA pot transforma managementul producției de serie, contribuind la eficientizarea proceselor și la adaptarea rapidă la cerințele în schimbare ale pieței. Astfel, se evidențiază impactul semnificativ pe care tehnologiile IA îl pot avea asupra performanței operaționale în industria producției de serie. [59]

2.1.1. Strategii de planificare agregată

Planificarea agregată, cunoscută și ca planificare pe termen mediu, este realizată după adoptarea deciziilor strategice privind produsele, structura comenzilor, tehnologiile, capacitățile de producție, politica de aprovizionare și vânzare.

Procesul de planificare agregată urmărește să determine cea mai avantajoasă cale de a satisface previziunile cererii prin ajustarea cantității de produse realizate, utilizarea forței de muncă în programe suplimentare, politica de stocuri, subcontractare sau alte variabile controlabile. [60] Încă din 1984, Prof. Dan Cândea și Prof. Ioan Abrudan defineau conceptul de planificare agregată a producției, descriind caracteristicile unui sistem de producție cu stadii multiple și modul de aplicare a programării liniare. [39]

Operatorii economici industriali actuali funcționează ca un sistem de "producție-stocuri" cu stadii multiple, unde produsul final se obține din mai multe piese, subansamble și ansamble componente. La finalul procesului tehnologic, materiile prime, materialele sau semifabricatele se transformă în produsul dorit de client. Fiecare etapă de transformare reprezintă un stadiu al procesului de fabricație. Literatura de specialitate acceptă conceptul conform căruia procesul decizional în activitatea de conducere este ierarhic, caracter imprimat și procesului de planificare al activităților productive. [61]

Planificarea unei firme se desfășoară în următoarele etape [62]:

1. **Planificare strategică sau de perspectivă** - Abordează strategiile de conducere privind direcțiile și investițiile pentru crearea sau extinderea obiectivelor economice, definirea strategiilor de marketing și asimilarea de produse noi în legătură cu direcțiile de specializare a producției.
2. **Planificarea tactică sau curentă** - Se concentrează pe alocarea și utilizarea eficientă a resurselor materiale, financiare și umane, conform deciziilor strategice implementate.
3. **Planificarea operativă sau programarea producției** - Detaliază sarcinile ce decurg din planul tactic, defalcate pe unități de timp scurte (ore, zile, săptămâni), până la nivel de executant direct.

Un element crucial al planificării agregate este utilizarea programării liniare pentru a optimiza utilizarea resurselor și a minimiza fluctuațiile. Programarea liniară este folosită pentru a determina planul optim de producție care minimizează costurile totale implicate. Planul de producție obținut în acest mod este denumit plan agregat, subliniind faptul că nu este un plan detaliat de fabricație, ci unul

care permite defalcarea volumului de producție pe produse individuale după stabilirea grupelor de produse. [63]

În cadrul unui sistem de producție complex, planificarea agregată trebuie să țină cont de diverse variabile și constrângeri care pot afecta procesul de producție. Aceste variabile includ cererea fluctuantă a pieței, disponibilitatea resurselor, capacitatea de producție și politica de stocuri. [64] De exemplu, în perioadele de cerere ridicată, este esențial să se ajusteze nivelul producției pentru a evita supraîncărcarea resurselor și creșterea costurilor. În schimb, în perioadele de cerere scăzută, este necesar să se reducă nivelul producției pentru a evita acumularea de stocuri și pierderile financiare.

În plus, planificarea agregată trebuie să integreze și politicile de subcontractare, care permit firmelor să externalizeze o parte din producție către terți. Aceasta poate fi o strategie eficientă pentru a gestiona fluctuațiile cererii și pentru a menține flexibilitatea în procesul de producție. Subcontractarea poate contribui, de asemenea, la reducerea costurilor fixe și la optimizarea utilizării capacităților interne de producție. [65]

Implementarea eficientă a planificării agregate necesită utilizarea unor instrumente și tehnici avansate de analiză și modelare, cum ar fi simulările și algoritmii de optimizare. Aceste instrumente ajută la evaluarea diferitelor scenarii de producție și la identificarea celor mai eficiente soluții pentru a satisface cerințele pieței și pentru a optimiza utilizarea resurselor. De asemenea, tehnologiile moderne, cum ar fi inteligența artificială și învățarea automată, pot juca un rol important în îmbunătățirea procesului de planificare agregată prin furnizarea de predicții precise și analize detaliate ale datelor. [66]

2.1.2. Tehnici de ordonanțare a producției

Ordonanțarea reprezintă o componentă esențială a programării producției, implicând eșalonarea în timp și spațiu a operațiilor de fabricație. Aceasta variază în funcție de tipul producției, fie că este de masă, de serie sau individuală. Pentru producția de masă, se elaborează un plan standard, pentru producția de serie un grafic coordonator, iar pentru producția individuală un grafic director. În fiecare tip de producție, ordonanțarea urmărește să încarce complet locurile de muncă și să minimizeze durata ciclului de fabricație, coordonând resursele necesare. [67]

Indiferent de tipul de producție, ordonanțarea prezintă dificultăți, atât în atingerea obiectivului (încărcarea completă a locurilor de muncă și minimizarea duratei ciclului de fabricație), cât și în

dirijarea resurselor necesare producției. Ordonanțarea producției de serie implica cele mai multe restricții. [68]

Aceste aspecte evidențiază complexitatea procesului de ordonanțare și necesitatea unei planificări riguroase pentru a asigura eficiența producției. Metodele tradiționale de ordonanțare includ planificarea după grafice fixe și ajustarea continuă a planurilor în funcție de evoluția producției. Cu toate acestea, avansurile în tehnologia informației și inteligența artificială au deschis noi oportunități pentru optimizarea ordonanțării. [69] [70]

Algoritmii genetici și alte tehnici de inteligență artificială oferă soluții avansate pentru problemele complexe de ordonanțare, permițând simularea și optimizarea fluxurilor de producție. Utilizarea acestor tehnici poate duce la îmbunătățirea semnificativă a eficienței operaționale, reducerea timpilor de așteptare și minimizarea costurilor de producție. [71]

De asemenea, integrarea sistemelor informatice avansate permite monitorizarea în timp real a proceselor de producție și ajustarea dinamică a planurilor de ordonanțare pentru a răspunde rapid la schimbările din mediu. [72] Acest nivel de flexibilitate și adaptabilitate este esențial pentru menținerea competitivității într-un mediu de afaceri din ce în ce mai dinamic și complex.

În concluzie, ordonanțarea producției este un proces esențial pentru asigurarea eficienței și competitivității în producția de serie.

2.1.3. Inovații în sistemele de planificare automată

Inovațiile în sistemele de planificare automată a producției reprezintă o componentă esențială în eficientizarea și modernizarea proceselor de producție de serie.

Prin utilizarea IA, sistemele pot procesa rapid o mare varietate de date - de la cerințele clienților până la starea echipamentelor - și pot lua decizii inteligente privind programarea producției. Acest lucru include optimizarea alocării resurselor, previziunea nevoilor de întreținere și adaptarea la schimbările cererii de piață în timp real. De exemplu, tehnologiile IoT permit monitorizarea continuă a proceselor de producție și colectarea de date în timp real, ceea ce îmbunătățește precizia planificării și permite o reacție rapidă la orice probleme operaționale.

Aceste inovații nu doar că sporesc eficiența și productivitatea, dar contribuie și la flexibilitatea și scalabilitatea producției. Sistemele de planificare automată moderne sunt esențiale pentru companiile care caută să se adapteze rapid la condițiile dinamice ale pieței și să rămână competitive în peisajul industrial contemporan. [73]

Un alt aspect al inovațiilor în planificarea automată este utilizarea algoritmilor de inteligență artificială, cum ar fi algoritmii genetici, optimizarea roiurilor de particule și algoritmii coloniilor de furnici. [74] Acești algoritmi permit optimizarea complexă a planificării, asigurând o alocare eficientă a resurselor și reducerea ciclurilor de producție. [75]

Algoritmii genetici, de exemplu, sunt capabili să identifice cele mai bune soluții prin simularea procesului de selecție naturală, în timp ce optimizarea roiurilor de particule și algoritmii coloniilor de furnici imită comportamentele naturale pentru a rezolva problemele de optimizare. [76] [77]

Sistemele de planificare automată utilizează, de asemenea, simulări și modele predictive pentru a îmbunătăți procesul decizional. Aceste simulări permit companiilor să testeze diferite scenarii și să identifice cele mai eficiente strategii înainte de a le implementa în practică. Modelele predictive ajută la anticiparea problemelor potențiale și la dezvoltarea unor planuri de acțiune care să minimizeze impactul acestora asupra producției. [78]

Integrarea acestor tehnologii avansate în sistemele de planificare automată permite nu doar optimizarea proceselor de producție, ci și îmbunătățirea comunicării și colaborării între diferitele departamente ale unei companii.

2.2. Bazele teoretice ale inteligenței artificiale

Inteligența artificială, definită ca abilitatea mașinilor de a imita procesele cognitive umane, precum învățarea și soluționarea problemelor, joacă un rol crucial în modernizarea și eficientizarea producției industriale. [79] Analiza istoriei și evoluției IA, de la originile sale în matematică și

filozofie până la dezvoltările recente în învățarea automată și rețelele neuronale, este esențială pentru a înțelege profunzimea acestei tehnologii.

IA a evoluat semnificativ, trecând de la conceptele teoretice la aplicațiile practice actuale, cum ar fi învățarea automată și rețelele neuronale. Aceste evoluții au condus la o diversificare a IA, în special în termeni de IA îngustă, concentrată pe sarcini specifice, și IA generală, care aspiră să simuleze inteligența umană generală. Înțelegerea acestor categorii este vitală pentru aplicarea IA în producția de serie, în special pentru automatizarea și optimizarea proceselor.

Tehnicile IA, cum ar fi învățarea supervizată, nesupervizată, învățarea prin întărire, rețelele neuronale și procesarea limbajului natural, sunt esențiale pentru dezvoltarea de algoritmi eficienți. În plus, metodele avansate de învățare profundă, cum ar fi rețelele neuronale convoluționale (CNN) și rețelele neuronale recurente (RNN), prelucrează și analizează cantități mari de date cu precizie și eficiență ridicate. [80] Aceste tehnologii au revoluționat domenii precum viziunea computerizată și recunoașterea vocală.

Fundamentele teoretice ale IA permit o evaluare riguroasă a potențialului și limitărilor acestei tehnologii în programarea producției. Analiza teoretică detaliată pregătește terenul pentru discuțiile ulterioare despre implementarea practică a IA în managementul producției de serie, subliniind rolul transformator al IA în industria modernă. Integrarea IA în procesele industriale îmbunătățește nu doar eficiența și calitatea producției, dar și permite o adaptabilitate sporită la cerințele dinamice ale pieței. [81]

De exemplu, IA facilitează personalizarea în masă, ajustând rapid liniile de producție pentru a răspunde preferințelor individuale ale clienților fără a compromite viteza sau costurile. Această flexibilitate este esențială într-un mediu de afaceri globalizat și competitiv, unde cerințele pieței se schimbă rapid. Integrarea IA în sistemele de producție permite, de asemenea, monitorizarea în timp real și adaptarea continuă a proceselor, asigurând astfel un nivel înalt de reactivitate și eficiență operațională.

2.2.1. Definiții ale inteligenței artificiale

Inteligența artificială este definită ca ramura științei computerelor dedicată creării de sisteme capabile să efectueze sarcini care necesită în mod tradițional inteligență umană. Aceasta include activități precum luarea deciziilor, rezolvarea de probleme, percepția, învățarea și adaptarea la noi contexte. Originea termenului de IA poate fi urmărită până la conferința de la Dartmouth din 1956, unde John McCarthy, adesea considerat părintele IA, a definit această disciplină ca *„știința și*

ingineria de a face mașini inteligente". [82] De atunci, IA a evoluat de la simple programe bazate pe reguli la sisteme complexe capabile de învățare și adaptare. **Definițiile clasice ale IA** includ abordarea lui John McCarthy, care a pus accentul pe crearea de sisteme care imită abilitățile cognitive umane.

Perspectiva funcțională. Alan Turing, unul dintre pionierii informaticii, a propus o viziune funcțională asupra IA prin faimosul său test Turing, unde o mașină este considerată „inteligentă" dacă poate imita comportamentul uman într-o manieră care să fie indiscernabilă de un om real. Această abordare a deschis discuția despre capacitatea mașinilor de a „gândi". [83]

Definiții bazate pe capacități. Stuart Russell și Peter Norvig, în lucrarea lor „Artificial Intelligence: A Modern Approach", clasifică IA în patru categorii: sisteme care gândesc ca oamenii, sisteme care acționează ca oameni, sisteme care gândesc rațional și sisteme care acționează rațional. Această clasificare subliniază diferența între imitarea procesului de gândire umană și realizarea acțiunilor raționale. [84]

Viziuni contemporane asupra IA. Autori contemporani accentuează rolul IA în învățare și adaptare. De exemplu, Yann LeCun, un pionier în învățarea profundă, definește IA ca fiind capacitatea unei mașini de a învăța din experiență și de a se adapta la noi situații. Această perspectivă evidențiază importanța învățării automate și a adaptabilității în dezvoltarea IA.

P.H. Winston, definește IA ca fiind „*studiul proceselor computaționale care permit percepție, gândire și acțiune.*" Această definiție subliniază aspectele esențiale ale IA, inclusiv capacitatea de a procesa informații senzoriale, de a raționa și de a lua decizii în mod autonom. [85]

2.2.2. Sistemul de clasificări al inteligenței artificiale (IA)

Inteligența artificială (IA) este un domeniu vast și diversificat, care poate fi clasificat în multiple moduri. Aceste clasificări ajută la înțelegerea complexității și varietății tehnicilor și aplicațiilor IA. În continuare, vom explora principalele categorii de clasificare ale IA [86]:

- *2.1.2.1. Clasificări ale inteligenței artificiale bazate pe lățime*

Clasificarea IA după lățime se referă la gama de abilități și aplicații ale sistemelor IA. Se face distincția între IA îngustă (sau slabă) și IA generală (sau puternică). IA îngustă este proiectată pentru a îndeplini sarcini specifice, cum ar fi recunoașterea facială sau analiza datelor, în timp ce IA generală are capacitatea de a efectua o sarcină intelectuală pe care o face un om.

- *2.1.2.2. Clasificarea inteligenței artificiale pe bază de modalități de învățare (IA simbolică și învățarea automată)*

Modalitățile de învățare în IA includ IA simbolică și învățarea automată. IA simbolică, predominantă în primii ani ai IA, se bazează pe utilizarea regulilor logice și a reprezentărilor simbolice pentru a efectua sarcini de calcul. Învățarea automată, pe de altă parte, permite sistemelor să învețe și să se îmbunătățească pe baza datelor. [87]

- *2.1.2.3. Clasificarea învățării automate după tipurile de aplicații*

Învățarea automată poate fi clasificată în funcție de tipurile de aplicații, cum ar fi clasificarea, regresia, clusteringul și reducerea dimensionalității. Clasificarea este utilizată pentru a atribui datele la categorii predefinite, regresia pentru a estima relațiile dintre variabile, clusteringul pentru a grupa datele similare, iar reducerea dimensionalității pentru a simplifica datele complexe păstrând caracteristicile esențiale [88]

- *2.1.2.4. Clasificarea învățării automate după modelele de învățare*

Modelele de învățare automată pot fi împărțite în modele deterministe și stocastice. Modelele deterministe oferă aceleași rezultate pentru aceleași seturi de date de intrare, în timp ce modelele stocastice includ un element de aleatoriu și pot produce rezultate diferite la rulări diferite. Această clasificare ajută la înțelegerea comportamentului algoritmilor de învățare și a aplicațiilor lor potrivite

- *2.1.2.5. Clasificarea învățării automate după paradigma algoritmilor*

Algoritmii de învățare automată sunt clasificați în funcție de paradigmele lor, incluzând algoritmi bazați pe arbori de decizie, rețele neuronale, metode de ansamblu și algoritmi genetici. Fiecare paradigmă are avantajele și dezavantajele sale, fiind aplicabile în diferite contexte și tipuri de date.

- *2.1.2.6. Clasificarea I.A. după adâncime (stratificare)*

IA poate fi clasificată și pe baza adâncimii sau stratificării, cunoscută sub numele de învățare profundă. Rețelele neuronale profunde (Deep Learning) folosesc multiple straturi de neuroni artificiali pentru a extrage caracteristici complexe din date. Această tehnologie a permis progrese semnificative în domenii precum recunoașterea imaginii și procesarea limbajului natural. [89]

- *2.1.2.7. Clasificarea rețelelor neurale artificiale pe baza algoritmilor*

Rețelele neurale artificiale (ANN) sunt clasificate în funcție de algoritmii utilizați pentru antrenare și funcționare. Acestea includ rețele neuronale convoluționale (CNN) pentru recunoașterea imaginilor, rețele neuronale recurente (RNN) pentru secvențe de date, și rețele generative adversariale (GAN) pentru generarea de date noi. Fiecare tip de rețea are aplicații specifice și avantaje distincte.

Aceste clasificări nu doar că oferă un cadru conceptual pentru organizarea cunoștințelor, dar și facilitează identificarea celor mai potrivite tehnici și algoritmi pentru probleme specifice.

În contextul tezei, importanța acestor clasificări este dublă. În primul rând, ele permit o analiză detaliată și structurată a diferitelor abordări și tehnologii din domeniul IA. Acest lucru este esențial pentru a putea evalua critic și a selecta cele mai eficiente metode pentru optimizarea proceselor de producție. De exemplu, învățarea automată și învățarea profundă sunt esențiale pentru dezvoltarea de algoritmi care pot analiza și interpreta date complexe de producție, anticipa cerințele viitoare și optimiza utilizarea resurselor.

În al doilea rând, aceste clasificări ajută la înțelegerea limitărilor și potențialelor fiecărei tehnologii. Cunoașterea diferențelor între IA îngustă și IA generală, de exemplu, poate ghida așteptările privind capacitățile actuale ale sistemelor IA și direcționa cercetările viitoare spre dezvoltarea de soluții mai avansate. În mod similar, clasificarea algoritmilor de învățare automată după paradigme, cum ar fi arborii de decizie sau rețelele neuronale, permite o selecție mai informată a instrumentelor potrivite pentru diverse aplicații industriale.

2.2.3. Tehnici de programare specifice IA

Înțelegerea programării logice și funcționale este crucială pentru a aprecia cum pot fi construite și rafinate modelele și algoritmii de inteligență artificială (IA). Aceste tehnici oferă metode robuste și eficiente pentru a aborda probleme complexe din producția de serie, cum ar fi optimizarea proceselor sau raționamentul automatizat pentru luarea deciziilor.

Programarea logică este utilizată extensiv în IA și este centrată pe utilizarea logicii formale pentru a exprima algoritmi. Limbajul Prolog este un exemplu definitoriu în acest domeniu, oferind un mediu puternic pentru programarea bazată pe reguli și raționamentul deductiv. În programarea logică, problemele sunt exprimate sub forma unor fapte și reguli, iar motorul de inferență al limbajului Prolog caută soluții prin aplicarea acestor reguli. Această abordare permite dezvoltarea unor sisteme care pot realiza raționamente complexe și pot găsi soluții optime într-un mod sistematic. CPLEX OPL este utilizat pentru a modela și rezolva probleme de optimizare, iar componenta de programare cu constrângeri este un exemplu clar de utilizare a paradigmei programării logice.

Programarea funcțională, reprezentată de limbaje precum LISP (List Processing) și, mai recent, de Jess și CLIPS, pune accentul pe calcularea rezultatelor prin aplicarea funcțiilor, spre deosebire de schimbarea stărilor. LISP a fost unul dintre primele limbaje de programare adoptate în cercetarea IA, datorită flexibilității sale și a capacității de a manipula simboluri și liste. Jess și CLIPS sunt exemple mai moderne, folosite pentru a dezvolta sisteme expert și pentru raționamentul bazat pe

reguli. Programarea funcțională se caracterizează prin utilizarea funcțiilor pure, care nu au efecte secundare, ceea ce facilitează verificarea corectitudinii și previzibilitatea comportamentului programelor.

Aceste limbaje de programare sunt esențiale pentru crearea de sisteme expert, procesarea limbajului natural și alte aplicații de IA care necesită abstracție și flexibilitate. Aceste tehnici permit dezvoltarea de soluții personalizate și optimizate pentru diverse probleme, contribuind semnificativ la eficientizarea și inovarea în producția de serie. Utilizând programarea logică și funcțională, cercetătorii pot crea sisteme IA care nu doar îmbunătățesc procesele existente, ci deschid noi posibilități pentru automatizarea și optimizarea proceselor industriale. Aceste tehnici sunt fundamentale pentru dezvoltarea algoritmilor care pot învăța și se pot adapta, permițând crearea unor soluții tot mai sofisticate și eficiente.

2.3. Evaluarea algoritmilor IA în programarea producției

Subcapitolul aduce în discuție eficacitatea și aplicabilitatea concretă a IA în contextul real al managementului producției de serie. Acest subcapitol face trecerea de la teoria IA și descrierea tehnologiilor sale (prezentate în secțiunile anterioare) la evaluarea practică și critică a impactului acestor tehnologii asupra proceselor de producție.

Acest subcapitol servește ca o punte vitală între cunoașterea teoretică a IA și implementarea practică în industria producției. Prin evaluarea algoritmilor de IA, se aduce o perspectivă practică

asupra modului în care tehnologiile IA pot fi folosite pentru a aborda provocările reale din domeniul producției.

De asemenea, se analizează modul în care algoritmii de IA îmbunătățesc procesele de producție, cum ar fi eficientizarea fluxurilor de lucru, reducerea timpilor de așteptare, creșterea calității produselor și optimizarea utilizării resurselor.

Subcapitolul nu doar că prezintă avantajele IA, dar oferă și o evaluare critică, abordând posibilele provocări și limitări ale aplicării IA în producție, cum ar fi complexitatea implementării, necesitățile de formare a angajaților și costurile asociate.

Importanța pentru industria producției de serie:

- **orientarea către inovație**: subliniază modul în care inovațiile în IA pot conduce la schimbări semnificative în industria producției, oferind soluții pentru creșterea eficienței și competitivității.
- **aplicabilitate extinsă**: analiza algoritmilor de IA în acest subcapitol este esențială pentru orice companie din domeniul producției de serie care aspiră să încorporeze tehnologii avansate pentru a-și îmbunătăți operațiunile.

Prin urmare, subcapitolul 2.3 constituie un element cheie al tezei, oferind nu doar o perspectivă valoroasă asupra potențialului IA în producție, ci și o abordare echilibrată și critică necesară pentru o înțelegere cuprinzătoare și aplicată a acestui domeniu dinamic și în continuă evoluție.

2.3.1. Rolul algoritmilor de optimizare

Rolul algoritmilor de optimizare este crucial în cadrul inteligenței artificiale computaționale (IAC), contribuind semnificativ la optimizarea proceselor de producție în serie prin diverse tehnici și algoritmi specifici. Printre aceste tehnici, Rețelele Neuronale Artificiale, Logica Fuzzy și Algoritmii Evolutivi sunt deosebit de relevante, fiecare aducând beneficii unice în contextul producției industriale.

Algoritmii Evolutivi constituie un subdomeniu cheie al IAC, inspirați de procesele biologice de evoluție și selecție naturală. Ei operează printr-un proces iterativ de generare și dezvoltare a unei populații de soluții, care sunt evaluate și selecționate în căutarea uneia optime. Acest proces este utilizat pentru a aborda probleme complexe de optimizare în producție, cum ar fi minimizarea timpului total de producție și maximizarea eficienței utilizării mașinilor. [90]

Algoritmi de Optimizare Metaeuristici - în cadrul acestor tehnici evolutive, se remarcă mai mulți algoritmi metaeuristici, fiecare având particularitățile sale [91]:

- **Algoritmi Genetici**: Utilizează mecanisme inspirate de genetică și selecție naturală pentru a genera soluții optime. Aceștia sunt eficienți în tratarea problemelor complexe cu spații de căutare mari.
- **Algoritmi Memetici**: Combină algoritmii genetici cu optimizarea locală, oferind un echilibru între explorarea globală și exploatarea locală.
- **Swarm Intelligence (Inteligența Roiurilor)**: Include algoritmi bazati pe comportamentele colective ale speciilor animale, cum ar fi [92]:
  - **Particle Swarm Optimization (Optimizarea pe Baza Roiurilor de Particule)**: Inspirat de comportamentul social al păsărilor sau peștilor, optimizează soluțiile prin actualizarea continuă a pozițiilor particulelor în spațiul de căutare. Particulele își ajustează pozițiile pe baza experienței lor și a celor mai bune soluții găsite de roi.
  - **Ant Colony Optimization (Optimizarea cu Colonii de Furnici)**: Bazat pe comportamentul furnicilor în căutarea hranei, utilizează feromonii pentru a ghida căutarea soluțiilor optime. Furnicile virtuale marchează traseele bune cu feromoni, iar aceste trasee sunt urmate de alte furnici, consolidând astfel căile eficiente. [93]
  - **Artificial Bee Colony Optimization (Optimizarea cu Roiuri de Albine)**: Simulează comportamentul inteligenței albinelor în căutarea hranei, optimizând soluțiile prin explorarea și exploatarea surselor de hrană. Albinele lucrătoare evaluează și partajează informații despre calitatea surselor de hrană, permițând roiului să se concentreze pe cele mai profitabile resurse.
  - **Artificial Immune System (Algoritmul Sistemului Imunitar Artificial)**: Inspirat de sistemul imunitar uman, acest algoritm detectează și elimină soluțiile necorespunzătoare, similar modului în care sistemul imunitar luptă împotriva patogenilor. Algoritmul utilizează concepte precum diversitatea și memoria imunologică pentru a îmbunătăți procesul de optimizare.

Cercetările în această zonă ([94] [95] ș.a.m.d.) s-au concentrat pe minimizarea ciclurilor de producție și pe eficientizarea utilizării echipamentelor, abordând provocări precum reducerea timpilor de așteptare și optimizarea secvențelor de operații. Prin aplicarea acestor algoritmi de optimizare,

producătorii pot atinge un nivel superior de eficiență și adaptabilitate, elemente esențiale în producția de serie modernă.

În plus față de tehnicile menționate anterior, algoritmii de optimizare joacă un rol crucial în diverse alte domenii de aplicare în cadrul producției. Un exemplu notabil este utilizarea lor în managementul lanțului de aprovizionare, unde optimizarea traseelor de transport și a stocurilor poate duce la economii semnificative și îmbunătățirea serviciilor. Algoritmii de optimizare sunt, de asemenea, esențiali în planificarea producției, ajutând la determinarea celor mai eficiente programe de producție pentru a satisface cererea variabilă a pieței. [96]

Un alt domeniu important este controlul proceselor industriale, unde algoritmii de optimizare sunt folosiți pentru a regla și ajusta parametrii operaționali în timp real, asigurând astfel calitatea constantă a produselor și minimizarea deșeurilor. În contextul mentenanței predictive, acești algoritmi pot analiza datele de performanță ale echipamentelor pentru a anticipa și preveni defecțiunile, contribuind la reducerea costurilor de întreținere și a timpilor de nefuncționare. [97]

Algoritmii de optimizare sunt, de asemenea, relevanți în domeniul designului de produs, unde pot ajuta la identificarea celor mai bune configurații și materiale pentru a îmbunătăți performanța și a reduce costurile. De exemplu, optimizarea topologică este o tehnică utilizată pentru a crea structuri eficiente din punct de vedere material, economisind resurse și reducând greutatea produselor finale.

2.3.2. Specificul tehnicilor IA computaționale

Subcapitolul 2.3.2 al tezei de doctorat abordează specificul tehnicilor de inteligență artificială (IA) computațională, concentrându-se pe diverse metode de optimizare care au aplicații semnificative în contextul programării producției.

2.3.2.1. Particle Swarm Optimization - se bazează pe comportamentul social al păsărilor sau peștilor. În programarea producției, PSO poate fi folosit pentru a optimiza distribuția resurselor și pentru a minimiza timpul de producție. Algoritmul ajustează pozițiile "particulelor" în spațiul soluțiilor, căutând continuu o soluție optimă prin urmărirea "celor mai bune" poziții înregistrate. [98]

2.3.2.2. Ant Colony Optimization - este inspirat de comportamentul furnicilor în căutarea hranei și este eficient în rezolvarea problemelor de optimizare combinatorie. În producție, ACO poate fi aplicat pentru a optimiza planificarea și ordonanțarea sarcinilor, utilizând "feromonii" artificiali pentru a găsi cele mai eficiente căi prin spațiul soluțiilor . [99]

2.3.2.3. Principiile Simulated Annealing - este o tehnică inspirată de procesul de călire în metalurgie. Aceasta abordează problema optimizării prin imitarea procesului de încălzire și răcire controlată pentru a atinge un stat de minimă energie (sau cost). SA este utilizat în producție pentru a găsi soluții apropiate de optim în contextul problemelor complexe de programare. [100]

2.3.2.4. Aplicații ale algoritmilor genetici (AG) - sunt inspirați de procesele naturale de evoluție și selecție. În contextul programării producției, AG sunt utilizați pentru a găsi soluții optime la probleme complexe, cum ar fi optimizarea traseelor de producție și alocarea resurselor. Prin simularea proceselor de selecție, încrucișare și mutație, AG pot explora un spațiu vast de soluții posibile, identificând configurații eficiente pentru linii de producție. [101]

O aplicare notabilă a AG în producție este optimizarea planificării producției. Aceasta implică determinarea secvențelor optime de operații pentru a minimiza timpii de oprire și a maximiza utilizarea echipamentelor. AG poate crea diverse secvențe de operații și poate evalua performanța fiecăreia, selectând cele mai eficiente pentru implementare. De asemenea, AG pot fi utilizati pentru optimizarea mixului de produse, determinând cea mai bună combinație de produse care să maximizeze profitul și să minimizeze costurile de producție.

În managementul lanțului de aprovizionare, AG sunt folosiți pentru a optimiza rutele de transport și pentru a minimiza costurile de logistică. Prin evaluarea diferitelor rute posibile și a costurilor asociate fiecărei rute, AG pot selecta cele mai eficiente trasee pentru transportul materiilor prime și al produselor finite. Acest lucru nu doar reduce costurile, dar îmbunătățește și timpii de livrare și satisfacția clienților.

AG sunt, de asemenea, aplicați în optimizarea configurării echipamentelor și a layout-ului fabricilor. Aceasta implică determinarea celei mai eficiente dispuneri a mașinilor și a stațiilor de lucru pentru a reduce timpul de transport al materialelor și a îmbunătăți fluxul de producție. Prin simularea și evaluarea diferitelor configurații, AG pot ajuta la identificarea designului optim al fabricii care să maximizeze eficiența operațională.

2.3.2.5. Algoritmii hibrizi (AG + PSO) – combină punctele forte ale diferitelor tehnici de optimizare pentru a obține soluții mai bune și mai rapide. În contextul producției, combinația dintre algoritmii genetici (AG) și Particle Swarm Optimization (PSO) aduce multiple beneficii. AG este eficient în explorarea globală a spațiului de soluții, dar poate suferi de convergență lentă. În schimb, PSO oferă o convergență rapidă prin ajustarea constantă a pozițiilor particulelor, însă poate cădea în capcana optimelor locale. [102]

Prin combinarea acestor două metode, se obțin avantaje semnificative. Algoritmii hibrizi pot începe cu o populație de soluții generate de AG, apoi folosesc PSO pentru a rafina aceste soluții,

îmbunătățind astfel atât viteza, cât și calitatea soluțiilor finale. În producția de serie, acești algoritmi hibrizi pot optimiza alocarea resurselor și programarea sarcinilor, reducând ciclurile de producție și crescând eficiența globală. [103]

Un exemplu de aplicare al algoritmilor hibrizi AG + PSO este în optimizarea liniilor de asamblare. AG generează o gamă largă de configurații posibile ale liniei de asamblare, explorând diverse combinații de secvențe de operații și alocări de resurse. Ulterior, PSO ajustează și rafinează aceste configurații, considerând factori dinamici precum schimbările în cererea de produse sau disponibilitatea echipamentelor. Această combinație permite găsirea rapidă a soluțiilor eficiente și adaptabile.

Algoritmii hibrizi pot fi utilizați și în managementul stocurilor și al aprovizionării. AG explorează strategii de aprovizionare și stocare, generând soluții care minimizează costurile și maximizează disponibilitatea produselor. PSO optimizează acești parametri, asigurând un echilibru optim între costuri și eficiență.

În planificarea producției, algoritmii hibrizi ajută la secvențierea sarcinilor pe echipamente și la alocarea resurselor umane. AG creează planuri de producție inițiale, optimizate apoi de PSO pentru a ține cont de constrângerile de timp și resurse. Această abordare asigură planuri de producție fezabile și optimizate pentru a maximiza producția și a minimiza întârzierile. [104]

Un avantaj al algoritmilor hibrizi AG + PSO este flexibilitatea și adaptabilitatea în fața perturbărilor. În caz de defecțiuni ale echipamentelor sau schimbări bruște ale cererii, aceștia recalculează rapid soluțiile optime, ajustând planurile de producție și alocarea resurselor pentru a menține eficiența operațională. Această capacitate de a răspunde rapid la schimbări este esențială în mediile de producție moderne, caracterizate de volatilitate și complexitate. [105] [106]

Alte tehnici IA computaționale

2.3.2.6. Deep Learning - sunt tehnici avansate de învățare automată care utilizează rețele neuronale profunde pentru a modela și a rezolva probleme complexe. În producție, deep learning poate fi folosit pentru predicții precise și analiza defectelor.

2.3.2.7. Reinforcement Learning (RL) – este o metodă în care agenții învață să ia decizii optimizând recompensele pe termen lung. În contextul producției, RL poate fi aplicat pentru optimizarea fluxurilor de lucru și a deciziilor de alocare a resurselor.

2.3.2.8. Sisteme Expert – sunt sisteme bazate pe cunoașterea domeniului specific și pe raționament bazat pe reguli. Acestea pot oferi asistență în luarea deciziilor complexe, cum ar fi diagnosticarea defecțiunilor și gestionarea mentenanței.

2.3.2.9. Case-Based Reasoning (CBR) – este o metodă care rezolvă probleme noi prin adaptarea soluțiilor de la probleme similare anterioare. În producție, CBR poate fi folosit pentru a adapta soluții de design sau procese în funcție de experiențele anterioare.

Fiecare dintre aceste tehnici IA computaționale oferă abordări unice și valoroase în optimizarea și eficientizarea programării producției, contribuind semnificativ la modernizarea și îmbunătățirea eficienței în producția industrială.

PARTEA APLICATIVĂ

CAPITOLUL 3. CERCETĂRI PRIVIND MODELAREA PROGRAMĂRII PRODUCȚIEI DE SERIE

3.1. Metodologia de cercetare aplicată

Lucrarea de cercetare a plecat de la ***întrebarea principală*****: În ce măsură contribuie aplicarea algoritmilor IA (genetici) și a altor tehnici de optimizare la îmbunătățirea procesului de programare a producției de serie?**

Răspunsul la întrebarea de cercetarea este: **În acest moment, metodele exacte sunt cele mai indicate în îmbunătățirea procesului de programare a producției de serie. Metodele euristice care folosesc algoritmi hibrizi specializați în optimizarea proceselor de planificare a producției au cunoscut o evoluție semnificativă, fără însă a furniza soluții identice cu cele ale metodelor exacte clasice.**

Cercetarea are următoarele **obiective principale:**

1. Identificarea elementelor noi care contribuie la îmbunătățirea procesului de planificare prin optimizarea proceselor de programare a producției de serie.

2. Optimizarea proceselor de producție prin tehnici avansate, mai exact investigarea contribuției algoritmilor genetici și a altor tehnici de optimizare la procesele de programare a producției.

3. Evaluarea aplicabilității IA în managementul producției, mai exact analiza modului în care diferite tipuri de algoritmi IA pot fi aplicați în programarea producției pentru a îmbunătăți eficiența și adaptabilitatea.

4. Identificarea principalelor tendințe în domeniul managementului programării producției de serie.

Pentru a pune în realizare obiectivele de cercetare și pentru a putea răspunde la întrebarea de cercetare au fost formulate următoarele ipoteze de cercetare:

Ipoteza 1. Metodele exacte oferă rezultate superioare celor euristice în programarea producției de serie.

Ipoteza 2. Algoritmii genetici hibrizi oferă rezultate superioare celor genetici clasici în ceea ce privește exactitatea soluțiilor oferite în programarea producției de serie.

Ipoteza 3. Algoritmii genetici clasici oferă rezultate diferite și neconcludente în rulările succesive în programarea producției de serie.

Ipoteza 4. Aplicarea modelului Guinet în programarea producției va conduce la îmbunătățiri semnificative în reducerea ciclurilor de producție.

Ipoteza 5. Rezultatele aplicării algoritmilor hibrizi au șanse ca pe viitor să se apropie de rezultatele aplicării metodelor exacte în programarea producției de serie.

Metodele de cercetare cantitative și calitative utilizate vor fi:

- analiza literaturii de specialitate - analiza datelor statistice obținute din surse secundare, inclusiv articole științifice, rapoarte industriale și studii de caz existente, pentru a identifica tendințele și evoluțiile recente în aplicarea algoritmilor IA și tehnicilor de optimizare în producția de serie.

- interviul ghidat - realizarea de interviuri cu experți din industrie și cercetători pentru a obține perspective aprofundate asupra aplicabilității și eficienței tehnicilor de optimizare și algoritmilor IA în programarea producției.

- analiza cauză – efect pentru a identifica și analiza factorii care contribuie la provocările și oportunitățile în programarea producției de serie, facilitând înțelegerea relațiilor complexe dintre diferitele elemente implicate.

- importanța studiului de caz- Identificarea și analiza unor studii de caz relevante pentru a demonstra aplicabilitatea practică a tehnicilor de optimizare și algoritmilor IA în diverse contexte industriale. Studiile de caz vor oferi o înțelegere practică a modului în care aceste tehnici pot fi implementate și adaptate în funcție de cerințele specifice ale fiecărei industrii.

- rezolvarea matematică a soluției problemei funcției obiectiv - Aplicarea metodelor exacte (CPLEX/OPL Mixt Integer Linear Programming și CPLEX Constraint Programming) și a metodelor euristice (algoritmul genetic AG și algoritmul hibrid AG + PSO) pentru a rezolva problemele de optimizare în programarea producției de serie. Aceasta implică formularea funcției obiectiv și a setului de constrângeri, urmată de analiza și compararea soluțiilor obținute.

- analiza SWOT - asociată aplicării algoritmilor IA și a tehnicilor de optimizare în programarea producției de serie. Aceasta va ajuta la dezvoltarea unor strategii informate și la luarea deciziilor strategice pentru îmbunătățirea performanței operaționale.

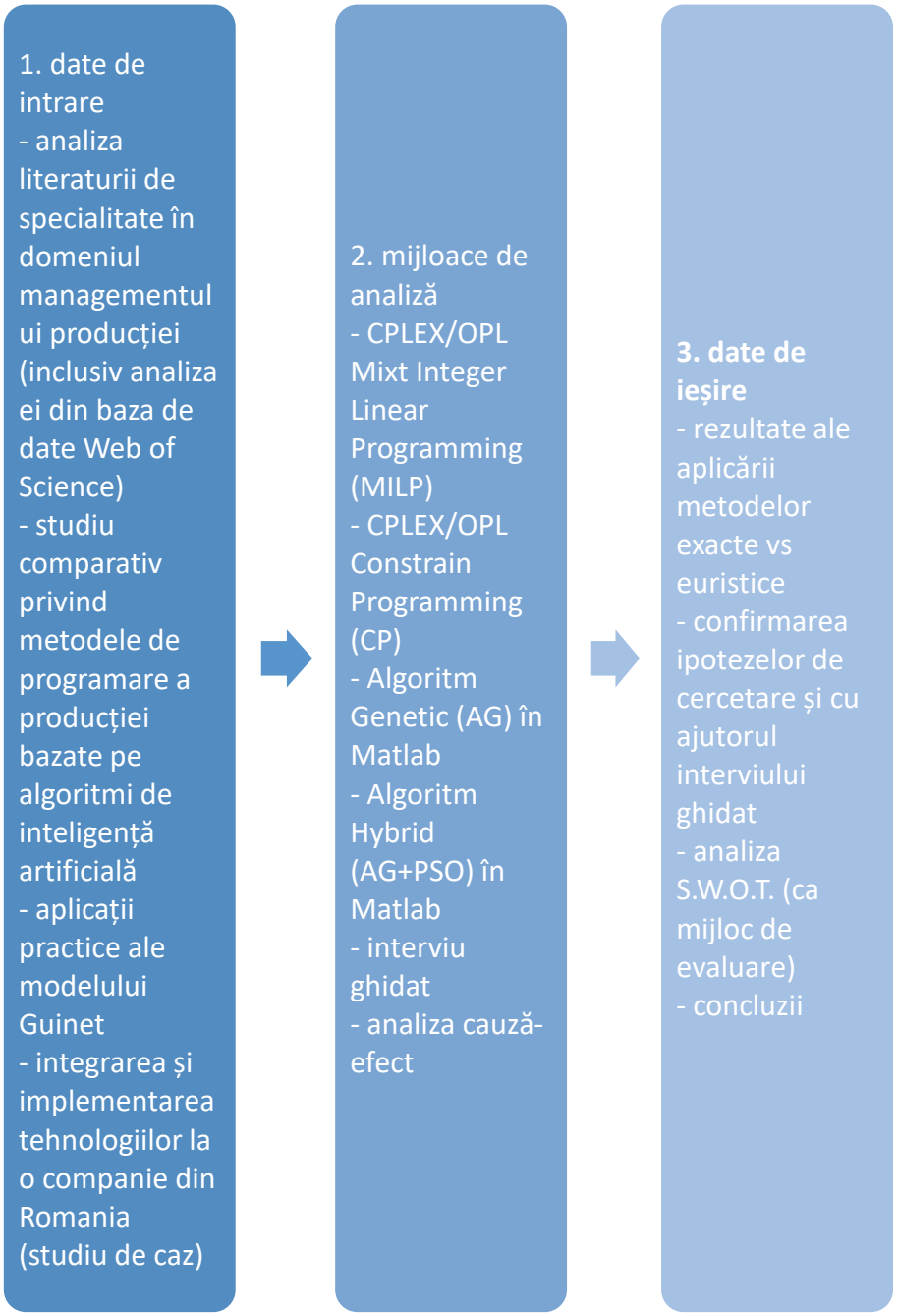


Figura 3.1. Metodologia de cercetare

(sursa: prelucrare proprie)

3.2. Prezentarea modelului Guinet și aplicațiile lui practice

Scopul funcției obiectiv, elaborată de Guinet și colegii săi, a fost minimizarea duratei celui mai lung ciclu de producție, realizând o structură simplă, descrisă în continuare [107] :

$$\sum_{i=0}^{n}\sum_{j=0}^{k_i} X_{i,j,k} = 1 \quad \forall j = 1,\dots,n; \quad \forall l = 1,\dots,m \qquad (1)$$

$$\sum_{j=0}^{n} X_{i,j,k} \leq 1 \quad \forall i = 0, \dots, n; \quad \forall k = 0, \dots, k_i; \quad \forall l = 1, \dots, L \tag{2}$$

$$\sum_{i=0}^{n} X_{i,h,j,k} - \sum_{h \neq i}^{n} X_{h,j,k} = 0 \quad \forall i = 0, \dots, n; \quad \forall k = 0, \dots, k_i; \quad \forall l = 1, \dots, L \tag{3}$$

$$C_j^{(l)} + \sum_{k=1}^{k_i} X_{i,j,k} \cdot P_{i,j} - \text{BigM}\left(1 - \sum_{k=1}^{k_i} X_{i,j,k}\right) \leq C_j^{(l)} \quad \forall i = 1, \dots, n; \quad \forall j = 1, \dots, m \tag{4}$$

$$C_j^{(l+1)} + P_{i,j} \leq C_j^{(l)} \quad \forall j = 1, \dots, n; \quad \forall l = 2, \dots, m \tag{5}$$

$$C_j^{(l)} \leq C_{\max} \quad \forall j = 1, \dots, n; \quad \forall l = 1, \dots, m \tag{6}$$

Unde:

• $X_{i,j,k,l}$ este o variabilă binară care are valoarea 1 atunci când comanda J_j este procesată imediat după comanda J_i pe mașina P_k în stadiul l;

• $X_{0,i,k,l}$ are valoare 1 atunci când comanda J_i este prima comandă programată pe mașina P_k în stadiul l;

- $X_{i,0,k,l}$ are valoarea 1 atunci când comanda J_i este ultima comandă programată pe mașina P_k în stadiul l;
- $C_j^{(1)}$ - indică timpul de încheiere a prelucrării comenzii J_j în stadiul l;
- • C_{max} – timpul de încheiere a tuturor comenzilor.

Modul de structurare al metodei elaborată de Guinet și colegii săi a avut ca finalitate:

• să obțină o soluție în care fiecare comandă este procesată o singură dată în fiecare stadiu al traseului urmat în sistemul de producție (constrângerile (1), (2) și (3));

• să asigure atribuirea de valori adecvate pentru variabilele $C_j^{(1)}$; prin constrângerea (4) modelul stabilește valorile variabilelor pentru două comenzi programate succesiv pe aceeași mașină, în timp ce constrângerea (5) impune modul de trecere a comenzilor prin stadii (în ordine crescătoare de la 1 la m); constrângerea (6) asigură atribuirea de valori variabilelor $C_j^{(1)}$mai mici decât C_{max}. [107]

Aplicații practice ale metodei Guinet

a) Gestionarea paturilor de spital

Metoda elaborată de Guinet și colegii săi a fost aplicată și pentru gestionarea paturilor de spital, problemă apărută din cauza reducerilor bugetare, care au impus organizarea optimă a resurselor

în funcție de cereri și cazurile acute, pentru a echilibra creșterea cererii cu resursele spitalicești limitate.

În cadrul unui studiu realizat în spitalul Sfântul Iosif/Sfântul Luc, s-a constatat că aceste unități, fiind sisteme complexe care gestionează un număr mare de resurse materiale și umane, necesită reducerea costurilor și menținerea competitivității, oferind cea mai bună îngrijire a pacienților. Spitalele acceptă atât pacienți electivi, cât și acuți, iar una dintre principalele probleme a fost gestionarea paturilor între aceste două categorii, având o cerere variabilă și imprevizibilă. Cum paturile sunt limitate, fiecare întârziere sau anulare afectează calitatea îngrijirii și profitul spitalului. [51]

Studiul a arătat că planificarea ocupării paturilor este complicată de constrângeri multiple (fără camere mixte, pacienți contagioși, incompatibilitate între patologii) și modificări frecvente din cauza duratei imprevizibile a ședерii și a necesității de a interna pacienți acuți.

Astfel, s-a propus un model decizional pentru managementul capacității paturilor, care să includă atât pacienții electivi, cât și pe cei acuți, respectând constrângerile legate de camere și patologii, bazându-se pe un model existent (Ben Bachouch și colaboratorii, 2007).

Abordarea a avut la bază un model existent (Ben Bachouch și colaboratorii, 2007) utilizat pentru a introduce pacienții acuți în programul de ocupare a patului. S-au avut în vedere: camere single și duble, incompatibilitate între patologii și pacienți contagioși.

Modelul propus este descris cu următoarele specificații:

Indici:

Set de pacienți indexați de $i \in P=1,...,N$

Set de zile indexate de $t \in D=1,...,T$

Set de paturi indexate de $l \in B=,...,L$

Date:

Debuți - prima perioadă de spitalizare a pacientului i;

Tard - ultima perioada de spitalizare a pacientului i;

S_i sexul pacientului i;

P_i Patologia a pacientului i;

C_i starea contagioasă a pacientului i;

L_oS_i durata șederii pacientului i;

T_2A_i Ratele de activitate pentru fiecare pacient;

H costul unei zile de întârziere privind spitalizarea pacientului;

HV o valoare ridicată;

Disponibilitate pat:

$$B_{lt}\begin{cases} -2 \text{ dacă patul l este deja atribuit unui bărbat în perioada t} \\ 1 \text{ dacă patul l este disponibil în perioada t} \\ 2 \text{ dacă patul 1 este deja atribuit unei femei în perioada t} \end{cases}$$

Amplasarea patului:

$$Mll'\begin{cases} 1 \text{ dacă paturile l și l'sunt în aceeași cameră dublă sau dacă patul l este localizat într} - \text{o cameră single } (l = l') \\ 0 \text{ în alte cazuri} \end{cases}$$

Sexul pacientilor:

$$Si\begin{cases} -1 \text{ dacă pacientul i este bărbat} \\ 1 \text{ dacă pacientul este femeie} \end{cases}$$

Disponibilitatea paturilor pe patologii:

$$PB_{lt}\begin{cases} Pi \text{ dacă patul l a fost atribuit anterior pacientului i care are patologia Pi în perioada t} \\ 0 \text{ dacă patul 1 este disponibil în perioada t} \end{cases}$$

Variabile binare:

Alocarea pacienților la paturi pentru fiecare perioadă:

$$X_{ilt}\begin{cases} 1 \text{ dacă pacientul i este repartizat în patul l în perioada t} \\ 0 \text{ în caz contrar} \end{cases}$$

Alocarea pacienților pe paturi:

$$A_{il}\begin{cases} 1 \text{ dacă pacientul i este repartizat la patul l} \\ 0 \text{ în alte situații} \end{cases}$$

Variabile întregi:

J_i perioada de început de spitalizare a pacientului i.

fin_i încheierea perioadei de spitalizare a pacientului i cum ar fi $fin_{i=}J_i+LoS_i-1$.

Variabila de transfer:

$$\text{REFUS}_i \begin{cases} 0 \text{ dacă pacientul i este repartizat la un pat} \\ L_o S_i \text{ altfel} \end{cases}$$

Problema de planificare a capacității patului ca un program liniar întreg a fost, așadar, formulată astfel [51]:

$$\text{Min} \sum_{i\in P}(J_i - \text{debut}_i)\cdot H + \sum_{i\in P}\left(\frac{\text{REFUS}_i}{\text{LoS}_i}\right)\cdot \text{T2A}_i \quad (1)$$

$$\forall i \in P, \forall t \in D: \sum_{l\in B} X_{ilt} \leq 1 \quad (2)$$

$$\forall i \in B, \forall t \in D: X_{ilt} \leq 1 \quad (3)$$

$$\forall i \in P: \sum_{l\in B}\sum_{t=\text{debut}_i}^{T} X_{ilt} + \text{REFUS}_i = \text{LoS}_i \quad (4)$$

$$\forall i \in P: \sum_{l\in B}\sum_{t=\text{debut}_i}^{T}\left(X_{ilt}\cdot(B_{lt}-2)\cdot(B_{lt}+2)\right)/(-3) = \text{LoS}_i \cdot A_{il} \quad (5)$$

$$\forall i \in P, \forall t \in D: J_i \leq \sum_{l\in B} t\cdot X_{ilt} + (-X_{ilt}+1)\cdot HV \quad (6)$$

$$\forall i \in P, \forall t \in D: f_{ini} \geq \sum_{l\in B} t\cdot X_{ilt} \quad (7)$$

$$\forall i \in P: f_{ini} = J_i + \text{LoS}_i - 1 \quad (8)$$

$$\forall i \in P: J_i \geq \text{debut}_i \quad (9)$$

$$\forall i \in P: J_i \leq \text{tard}_i \quad (10)$$

$$\forall l, l' \in B, l \neq l', \forall t \in D, \forall M_{l'} = 1: \sum_{i\in P/C_i=2} X_{ilt}\cdot S_i - \sum_{i\in P/C_i=2} X_{il't}\cdot S_i \leq 1 \quad (11)$$

$$\forall l, l' \in B, l \neq l', \forall t \in D, \forall M_{l'} = 1: \sum_{i\in P/C_i=2} X_{ilt}\cdot S_i - \sum_{i\in P/C_i=2} X_{il't}\cdot S_i \geq -1 \quad (12)$$

$$\forall l, l' \in B, l \neq l', \forall t \in D, \forall M_{l'} = 1: B_{lt}\cdot \sum_{i\in P/C_i=2} S_i \cdot X_{ilt} \leq 2 \quad (13)$$

$$\forall l, l' \in B, l \neq l', \forall t \in D, \forall M_{l'} = 1: B_{lt}\cdot \sum_{i\in P/C_i=2} S_i \cdot X_{il't} \geq -1 \quad (14)$$

$$\forall l, l' \in B, l \neq l', \forall t \in D, \forall M_{l'} = 1:$$

$$\sum_{i\in P/C_i=2} X_{ilt}\cdot P_i - \sum_{i\in P/C_i=2} X_{il't}\cdot P_i \leq 1 + \left(1 - \sum_{i\in P/C_i=2} X_{ilt}\right)\cdot HV \quad (15)$$

$$\forall l, l' \in B, l \neq l', \forall t \in D, \forall M_{l'} = 1:$$

$$\sum_{i\in P/C_i=2} X_{ilt}\cdot P_i - \sum_{i\in P/C_i=2} X_{il't}\cdot P_i \geq -1 - \left(1 - \sum_{i\in P/C_i=2} X_{ilt}\right)\cdot HV \quad (16)$$

$$\forall l, l' \in B, l \neq l', \forall t \in D, \forall M_{l'} = 1:$$

$$PB_{lt} - \sum_{i\in P/C_i=2} P_i \cdot X_{ilt} \leq 1 + \left(1 - \sum_{i\in P/C_i=2} X_{ilt}\right)\cdot HV \quad (17)$$

$$\forall l, l' \in B, l \neq l', \forall t \in D, \forall M_{l'} = 1:$$

$$PB_{lt} - \sum_{i \in P/C_i=2} P_i \cdot X_{il't} \geq -1 - \left(1 - \sum_{i \in P/C_i=2} X_{ilt}\right) \cdot HV \quad (18)$$

$\forall l, l' \in B, l \neq l', \forall t \in D, \forall M_{l'} = 1$:

$$B_{lt} \cdot \left(\sum_{i \in P/C_i=1} X_{ilt} + \sum_{i \in P} X_{il't}\right) \leq 1 \quad (19)$$

$\forall l, l' \in B, l \neq l', \forall t \in D, \forall M_{l'} = 1$:

$$B_{lt} \cdot \left(\sum_{i \in P/C_i=1} X_{ilt} + \sum_{i \in P} X_{il't}\right) \geq 0 \quad (20)$$

$$\forall i \in P, \forall l \in B, \forall t \in D: X_{ilt} \in \{0,1\} \quad (21)$$

$$\forall i \in P, \forall l \in B: A_{il} \in \{0,1\} \quad (22)$$

$$\forall i \in P: \text{REFUS}_i \geq 0 \quad (23)$$

$$\forall i \in P: J_i \geq 0 \quad (24)$$

$$\forall i \in P: f_{ini} \geq 0 \quad (25)$$

S-a concluzionat, prin această metodă, că funcția obiectiv minimizează două costuri: costul generat de întârzierea spitalizării pacientului (diferența dintre începutul spitalizării și perioada cea mai timpurie de spitalizare, înmulțit cu un cost fix) și costul unei admiteri refuzate din cauza indisponibilității patului. Fiecare spitalizare este asociată cu o plată definită pe baza activității T2A: tarifarea activității care induce o pierdere dacă ședera pacientului nu este programată corect.

S-a utilizat același cost pentru fiecare zi de întârziere, plecând de la premisa că paturile rămân disponibile indiferent de patologia asociată cu întârzierea. Acest cost este egal cu venitul mediu pierdut dintr-o zi suplimentară, în raport cu T2A.

Finalitatea studiului a constat în oferirea unui instrument pentru unitățile medicale, care să permită întocmirea unui program de ocupare a patului, ținând cont de pacienții acuți și electivi, de constrângerile camerelor, pacienții contagioși și incompatibilitatea între patologii.

Identificarea unei metode adecvate pentru acest model depinde de criterii variate. Pentru un număr mare de pacienți, CPLEX este cel mai potrivit, dar pentru un număr redus, atât LINGO, cât și CPLEX sunt eficiente. Studiul a arătat că metoda GLPK nu este adecvată pentru acest model, deși a fost eficientă în alte studii. Alegerea se reduce, așadar, la CPLEX și LINGO, deși pentru LINGO timpul de calcul crește exponențial odată cu numărul pacienților.

b) Trasabilitatea materiilor prime în industria alimentară

Multiplele crize de siguranță alimentară, cum ar fi ESB sau febra aftoasă, au determinat companiile alimentare să încerce să limiteze riscurile asumate și să liniștească consumatorii, scopul nefiind doar acela de a urmări produsele în mod eficient, ci și de a minimiza retragerile și numărul de loturi care constituie un anumit produs finit, considerent pentru care a fost propus un model matematic MILP și rezultatele experimentelor obținute cu software-ul LINGO.

Studiul elaborat în această direcție a tranșat ideea că, un bun sistem de trasabilitate stabilește cu exactitate istoricul compoziției și locației produselor de-a lungul lanțului de aprovizionare, în literatura de specialitate trasabilitatea fiind abordată din punctul de vedere al calității, modelării sau sistemului informațional. Autorii studiului au propus o nouă abordare prin îmbunătățirea trasabilității, abordare care încearcă să controleze amestecarea loturilor de producție pentru a limita dimensiunea și, în consecință, costul și impactul media al loturilor.

Astfel după cum au arătat Kim și colaboratorii (1995), un sistem de trasabilitate trebuie să poată urmări istoricul produselor și activităților, adică TRU (*en. Tracking Reference Units* – ro. unități de referință pentru transabilitate) și activitățile primare. Aceștia au definit câteva reguli fundamentale pentru trasabilitate, folosind modelul semantic al ontologiei lor și logica de ordinul întâi:

Dacă un TRU este divizat, părțile separate păstrează identificarea TRU-ului părinte.

Dacă unele TRU sunt asamblate, identificarea noului TRU este diferită de identificarea părintelui TRU-uri. [108]

Interesele înființării unui sistem de trasabilitate eficient în industria alimentară au fost evidențiate în marketing (pentru a asigura consumatorul cu etichete de calitate obținute cu un sistem de trasabilitate eficient), în producția produselor vândute cu o marcă de retailer, dar și pentru câștigarea de contracte prin întărirea credibilității producătorului (viteza de reacție, identificarea precisă a produselor).

Un sistem de trasabilitate bun defintește calitatea companiei prin urmărirea produselor, proceselor de producție și controalelor de calitate, chiar dacă prin sine nu îmbunătățește calitatea produselor, poate eficientiza respectarea legislației și reactivitatea la viitoarele legi, poate conduce la evitarea repetărilor inutile ale măsurilor asupra produselor (măsurile efectuate asupra componentelor nefiind necesare pentru subproduse în ipoteza în care loturile de producție sunt urmărite eficient).

Avantajele înființării trasabilității interne în companiile de producție a fost prezentată și de Moe (1998) și identificate ca:

- Posibilitatea de a crește controlul producției.
- Indicații pentru a găsi relația dintre cauză și efect în cazul produselor neconforme.
- Limitarea costului în cazul amestecului de produse de bună și proastă calitate.

• Ușurința de a găsi informații pentru un audit de calitate.

• Ușurința înființării sistemelor informaționale (managementul producției, stocuri, calitate etc).

Ca dezavantaj, s-a reținut că, deseori este dificil de evaluat rentabilitatea investiției, chiar în ciuda unui sistem de trasabilitate eficient, instituirea acestuia găsindu-și rațiunea în caz de criză a siguranței alimentare. Acesta nu poate reduce probabilitatea unei crize de siguranță alimentară, însă are menirea de a diminua consecințele acesteia.

Una dintre problemele vitale în caz de criză și, implicit, trei calități de bază ale unui sistem de trasabilitate eficient este reacția companiei care trebuie să fie rapidă, precisă și fiabilă.

În cazul unei crize de siguranță alimentară, au fost identificate interese ale unui sistem de trasabilitate eficient, redate în continuare [108]:

• Reducerea costurilor (de timp și personal) pentru căutarea istoricului și localizarea produselor în caz de probleme.

• Reducerea costurilor pentru rechemarea produselor: există mai puține produse de rechemat dacă sunt identificate, nevoia de a rechema produse deja procesate (sau chiar mai rău, distribuite clientului) este în cele din urmă redusă, iar numărul de clienți în cauză scade.

• Reducerea numărului de mărci sau site-uri de producție vizate de o rechemare pentru o companie cu mai multe site-uri sau mai multe mărci.

• Reducerea pierderii încrederii consumatorului în cazul unei probleme grave de siguranță alimentară, arătând că problema este sub control.

Efectuând o analogie între problema studiată și o problemă de transbordare cu costuri fixe, s-a concluzionat că sursele reprezintă loturile de materie primă, nodurile tranzitorii, loturile de componente și destinațiile loturilor de produs finit. Un arc modelează o legătură de dezasamblare sau de asamblare a unei liste de materiale, acesteia atribuindu-i-se un cost pentru fiecare utilizare, cost independent de curgerea arcului, adică fix. Procedând astfel, se urmărește ca suma legăturilor dintre loturile de materii prime și loturile de produse finite (adică suma costurilor fixe) să fie redusă la minimum.

Autorii studiului au propus un model matematic problemei de dispersie în care i, j, k și l sunt indici ai loturilor de materii prime, loturi de componente, loturi de produse finite și loturi de componente cumpărate.

Funcția obiectiv astfel elaborată permite calcularea dispersiei minime a lotului și a fost definită ca fiind suma legăturilor dintre loturile de materie primă și loturile de produs finit date de Y(i, k) și dispersia datorată componentelor cumpărate xBF(l, k).

S-a explicat că, în procesul de fabricație, cantitatea trebuie conservată. Constrângerile arată că totalul cantității limitate a unui lot de materie primă este utilizat în loturile de componente, iar totalul cantității unui lot de componente provine doar din loturi de materie primă. Fiecare lot de produs finit provine din loturi de componente și/sau din loturi de componente cumpărate; cantitățile acestora trebuie, de asemenea, păstrate, iar fiecare lot de componente este asamblat în întregime în loturi de produs finit, toate acestea fiind cuprinse în ecuațiile 4-8 din modelul matematic propus. [108]

Ecuațiile 8-10 exprimă faptul că variabilele binare xRC, xCF și xBF sunt egale cu 1 dacă QRC, QCF și QBF nu sunt nule. Ecuațiile 11 sunt utilizate pentru a determina Y(i, k), care este egal cu 1 dacă lotul de materie primă i este utilizat în lotul de produs finit k. Y(i, k) nu este definită ca o variabilă binară, dar fiind minimizată în funcția obiectiv, va lua automat valoarea 1 sau 0. Dacă atât xRC, cât și xCF sunt egale cu 1, Y(i, k) va fi 1; altfel, va fi 0, datorită funcției.

Data:

- **BoMRC (a, b)**: proporție de componente de tip b dată de o materie primă de tip a; aceasta este lista de materiale de demontare.
- **BoMCF (b, c)**: proporție de componente de tip b utilizată într-un produs finit de tip c; aceasta este lista de materiale de asamblare.
- **TRM (i)**: tipul lotului de materie primă i.
- **TFP (k)**: tipul lotului de produs finit k.
- **QRM (i)**: cantitatea lotului de materie primă i.
- **CFM (k)**: cantitatea lotului de produs finit k.
- **TCOMP (j)**: tipul lotului de componente j.
- **M.**: numărul de loturi de materii prime.
- **Nu.**: numărul de loturi de componente.
- **P.**: numărul de loturi de produs finit.
- **Q.**: numărul de loturi de componente cumpărate.
- **S.**: numărul de tipuri diferite de componente.
- **Vhv**: Valoare foarte mare.

Variabile:

- **Y (i, k)**: variabilă egală cu 1 dacă lotul de materie primă i este utilizat în lotul de produs finit k și 0 în caz contrar.

- **xBF (l, k)**: variabilă binară egală cu 1 dacă lotul de componente cumpărat l este utilizat în lotul de produs finit k și 0 în caz contrar.
- **xRC (i, j)**: variabilă binară egală cu 1 dacă lotul de materie primă i este utilizat în lotul de componente j și 0 în caz contrar.
- **xCF (j, k)**: variabilă binară egală cu 1 dacă lotul de componente j este utilizat în lotul de produs finit k și 0 în caz contrar.
- **QRC (i, j)**: variabilă care este cantitatea lotului de materie primă i utilizată în lotul de componente j.
- **QBF (l, k)**: variabilă care este cantitatea lotului de componente cumpărate l utilizată în lotul de produs finit k.
- **QCF (j, k)**: variabilă care este cantitatea lotului de componente j utilizată în lotul de produs finit k.
- **QCOMP (j)**: variabilă care este cantitatea lotului de componente j.

Modelul matematic propus:

$$\text{Minimize } Z = \sum_{i=1}^{M}\sum_{j=1}^{N} Y(i,j) + \sum_{k=1}^{P}\sum_{l=1}^{Q} X_{BF}(l,k) \quad (1)$$

$$BoM_{RC}\left(T_{RM}(i,b) \times Q_{RM}(i)\right) = \sum_{j=1}^{N} Q_{RC}(i,j) \quad \forall i = 1,\dots,M, \quad \forall b = 1,\dots,S \quad (2)$$

$$BoM_{CF}\left(b, T_{FP}(k)\right) \times Q_{FP}(k) = \sum_{j=1}^{N} Q_{CF}(j,k) + \sum_{l=1}^{Q} Q_{BF}(l,k) \quad \forall k = 1,\dots,P, \quad \forall b = 1,\dots,S \quad (3)$$

$$Q_{COMP}(j) = \sum_{i=1}^{M} Q_{RC}(i,j) \quad \forall j = 1,\dots,N \quad (4)$$

$$Q_{FP}(k) = \sum_{j=1}^{N} Q_{CF}(j,k) + \sum_{l=1}^{Q} Q_{BF}(l,k) \quad \forall k = 1,\dots,P \quad (5)$$

$$\sum_{k=1}^{P} Q_{CF}(j,k) = Q_{COMP}(j) \quad \forall j = 1,\dots,N \quad (6)$$

$$\sum_{j=1}^{N} Q_{RC}(i,j) = Q_{RM}(i) \quad \forall i = 1,\dots,M \quad (7)$$

$$x_{RC}(i,j) \le Q_{RC}(i,j)\backslash$$
$$Q_{RC}(i,j) \le x_{RC}(i,j) \times Vhv\backslash$$
$$\forall i = 1,\dots,M, \quad \forall j = 1,\dots,N \quad (8)$$

$$x_{CF}(j,k) \le Q_{CF}(j,k)\backslash$$
$$Q_{CF}(j,k) \le x_{CF}(j,k) \times Vhv\backslash$$
$$\forall k = 1,\dots,P, \quad \forall j = 1,\dots,N \quad (9)$$

$$x_{BF}(l,k) \le Q_{BF}(l,k)\backslash$$

$$Q_{BF}(l,k) \le x_{BF}(l,k) \times Vhv\backslash$$
$$\forall l = 1,\dots,Q, \quad \forall k = 1,\dots,P \tag{10}$$
$$x_{RC}(i,j) + x_{CF}(j,k) + Y(i,j,k) \le Y(i,j,k) + 1 \quad \forall i = 1,\dots,M, \quad \forall j = 1,\dots,N, \quad \forall k = 1,\dots,P \tag{11}$$

Cu modelul matematic propus, devine facilă determinarea dispersiei descendentă a unui lot de materie primă sau dispersia ascendentă a unui lot de produs finit prin ecuațiile 12 și 13. Pentru a se putea cunoșate dispersia descendentă a unui anumit lot de materie primă dacă, de exemplu, această materie primă prezintă o frecvență ridicată a problemelor de calitate, autorii studiului au propus următoarea ecuație [108]:

$$\text{D\_DISP}(i) = \sum_{k=1}^{P} Y(i,k) \tag{12}$$
$$\text{U\_DISP}(k) = \sum_{i=1}^{M} Y(i,k) + \sum_{l=1}^{Q} x_{BF}(l,k) \tag{13}$$

Pezentând perspectivele studiului, s-a arătat că situația particulară studiată a fost întâlnită în industria mezelurilor, modelul matematic propus pentru a reduce dispersia lotului fiind, însă, prea mare pentru a fi folosit zilnic în industrie, dar putând fi folosit cu modele simplificate.

Cercetări ulterioare pot fi efectuate, deoarece modelul este limitat, iar atunci când dimensiunea problemei crește, devine imposibil de utilizat. S-a trasat, ca posibilă direcție pentru cercetările viitoare, dezvoltarea unui algoritm euristic care ar putea rezolva problema într-un timp rezonabil.

Totodată, o posibilă rezolvare a cazului studiat ar putea fi elaborată și fără a fi necesară reducerea numărului de variabile.

Ținând cont de aspectul că ipoteza studiată este caracterizată de o listă de materiale pe 3 niveluri (materii prime, componente și produse finite), modelul de dispersie propus ar putea fi completat prin adăugarea unui al patrulea nivel, având în vedere procesul de ambalare. Un anumit lot de produs poate fi ambalat în multe moduri diferite; un anumit produs ambalat poate fi compus din diferite loturi de produse, se poate dezvolta și ipoteza, dacă modelul și rezultatele ar fi foarte diferite, cu un model de dispersie cu 4 straturi.

Formularea MILP are ca scop minimizarea dispersiei lotului, respectiv dimensiunea retragerilor de loturi în caz de criză a siguranței alimentare. Deci, cantitățile de materii prime și produse finite ar putea apărea în funcția obiectiv folosind dispersii în sus și în jos, conform ecuațiilor prezentate la punctele 12 și 13 din modelul matematic expus în precedent.

3.3. Interviuri ghidate cu specialiști din domeniu

3.3.1. Ghidul de interviu și importanța lui pentru tematica cercetării

Pentru a evalua importanța pe care specialiștii din industrie o acordă implementării tehnologiilor de inteligență artificială (IA) în producție, a fost realizat un interviu structurat cu 10 subiecți, manageri de nivel înalt din companii ce activează în diverse domenii. Aceștia au fost selectați pe baza disponibilității lor și au fost invitați să răspundă la o serie de întrebări relevante pentru ipotezele cercetării. În continuare, vom analiza răspunsurile obținute din aceste interviuri, corelându-le cu ipotezele formulate. Ghidul de interviu și răspunsurile primitese regăsește în Anexa 1.

1. Sunteți interesați să folosiți inteligența artificială în optimizarea programării producției?

Această întrebare este esențială pentru a evalua nivelul de interes și deschidere al companiilor față de tehnologiile IA în optimizarea producției. Este important să se determine dacă există un potențial semnificativ de implementare a acestor tehnologii inovative. Un interes ridicat poate indica

o pregătire adecvată pentru adoptarea IA, în timp ce un interes scăzut poate sugera necesitatea educării și informării suplimentare despre beneficiile IA. De exemplu, companiile care au integrat IA în procesele lor de producție au raportat creșteri semnificative ale eficienței și reduceri ale costurilor operaționale. Aceasta ar putea confirma Ipoteza 5, conform căreia algoritmii genetici hibrizi au șanse ca pe viitor să se apropie de rezultatele aplicării metodelor exacte în programarea producției de serie mare.

2. În ce măsură considerați că ar fi acceptat gradul de implementare a IA de către angajații companiei?

Această întrebare este crucială pentru a înțelege atitudinea angajaților față de adoptarea noilor tehnologii. Acceptarea de către angajați este vitală pentru succesul oricărei implementări tehnologice. Dacă angajații sunt reticenți sau nesiguri, acest lucru poate reprezenta un obstacol major. Companiile trebuie să fie pregătite să ofere programe de formare și suport pentru a facilita tranziția. Studiile arată că angajații bine informați și instruiți sunt mai predispuși să accepte și să adopte tehnologii noi. Acest lucru este relevant pentru Ipoteza 4, care sugerează că aplicarea modelului Guinet în programarea producției va conduce la îmbunătățiri semnificative în reducerea ciclurilor de productie.

3. Care este opinia dvs. cu privire la investiția în dezvoltarea softurilor care folosesc IA în programarea producției?

Evaluarea perspectivei companiei asupra investițiilor în software IA este crucială pentru a înțelege disponibilitatea de a aloca resurse financiare pentru aceste tehnologii. O abordare pozitivă față de investiții poate indica o dorință de modernizare și de menținere a competitivității pe piață. Pe de altă parte, reticența la investiții poate sugera o necesitate de a demonstra beneficiile tangibile ale IA în termeni de rentabilitate a investiției (ROI). Conform unui raport PwC, companiile care investesc în IA pot vedea o creștere de până la 20% în productivitate [109]. Aceasta poate susține Ipoteza 1, conform căreia metodele exacte oferă rezultate superioare celor euristice în programarea producției de serie.

4. Utilizați algoritmi de IA în programarea producției în acest moment (la momentul actual)?

Această întrebare ajută la determinarea stadiului actual al utilizării IA în companie. Dacă algoritmii IA sunt deja în uz, se poate discuta despre optimizarea și extinderea acestora. Dacă nu sunt utilizați, se pot identifica nevoile specifice și pașii necesari pentru implementare. Utilizarea actuală a

IA indică un nivel de încredere și familiaritate cu tehnologia, facilitând astfel adoptarea extinsă. Aceasta ar putea confirma Ipoteza 2, care sugerează că algoritmii genetici hibrizi oferă rezultate superioare celor genetici clasici în ceea ce privește exactitatea soluțiilor oferite în programarea producției de serie.

5. Ce capacități/abilități trebuie să aibă angajații care lucrează pentru implementarea IA în programarea producției?

Identificarea competențelor necesare pentru implementarea IA este esențială pentru asigurarea succesului proiectului. Angajații trebuie să posede abilități tehnice solide, cunoștințe de programare și o înțelegere profundă a proceselor de producție. În plus, competențele de analiză a datelor și învățare automată sunt vitale. Companiile trebuie să investească în formarea continuă a angajaților pentru a se asigura că aceștia pot utiliza eficient noile tehnologii. Aceasta susține Ipoteza 1, conform căreia metodele exacte oferă rezultate superioare celor euristice în programarea producției de serie.

6. Care sunt factorii care facilitează implementarea IA în programarea producției în compania în care lucrați?

Identificarea factorilor facilitatori este crucială pentru valorificarea avantajelor existente și dezvoltarea de strategii eficiente de implementare. Acești factori pot include suportul managerial, infrastructura tehnologică adecvată, cultura organizațională deschisă la inovare și disponibilitatea resurselor financiare. Un mediu de lucru colaborativ și deschis la schimbare este, de asemenea, esențial. Conform unui raport Gartner, companiile care beneficiază de suport managerial puternic și o cultură organizațională favorabilă inovației sunt mai susceptibile de a implementa cu succes tehnologii avansate precum IA. [110] Aceasta poate susține Ipoteza 4, conform căreia aplicarea modelului Guinet în programarea producției va conduce la îmbunătățiri semnificative în reducerea ciclurilor de productie.

7. Considerați că la momentul actual putem vorbi de o eficiență semnificativ îmbunătățită, inclusiv reducerea timpilor de așteptare, prin utilizarea de IA în managementul producției de serie?

Această întrebare este esențială pentru evaluarea percepției asupra eficienței actuale și a potențialului de îmbunătățire prin IA. Dacă IA poate reduce semnificativ timpii de așteptare și poate crește eficiența, aceasta ar oferi un avantaj competitiv considerabil. Studiile arată că utilizarea IA în producție poate duce la reducerea timpilor de oprire a mașinilor și la optimizarea programării,

contribuind la o producție mai fluidă și mai eficientă. [111] Acest lucru poate confirma Ipoteza 1, conform căreia metodele exacte oferă rezultate superioare celor euristice în programarea producției de serie.

8. Care sunt factorii care împiedică/încetinesc implementarea IA în programarea producției în compania în care lucrați?

Identificarea obstacolelor este crucială pentru dezvoltarea de soluții eficiente și facilitarea implementării IA. Barierele pot include lipsa resurselor financiare, reticența angajaților la schimbare, infrastructura tehnologică insuficientă și lipsa competențelor necesare. Înțelegerea acestor factori permite companiei să abordeze problemele și să dezvolte strategii pentru a le depăși. Conform unui studiu realizat de McKinsey, principalele bariere în implementarea IA sunt legate de competențele angajaților și infrastructura tehnologică. [112] Acest lucru poate susține Ipoteza 3, care sugerează că algoritmii genetici clasici oferă rezultate diferite și neconcludente în programarea producției de serie mare.

9. Cum ați descrie relația pe care o aveți/ați avea-o cu managerii din celelalte departamente în ceea ce privește implementarea IA în programarea producției?

Colaborarea interdepartamentală este esențială pentru succesul implementării IA. O relație bună cu managerii din celelalte departamente poate facilita integrarea IA în procesele de producție și asigura o comunicare eficientă. Este important ca toate părțile implicate să înțeleagă beneficiile IA și să colaboreze pentru a depăși eventualele provocări. Conform unui raport realizat de BCG, companiile care promovează o cultură de colaborare și comunicare deschisă între departamente sunt mai susceptibile de a implementa cu succes tehnologii avansate precum IA. [113] Aceasta poate susține Ipoteza 4, conform căreia aplicarea modelului Guinet în programarea producției va conduce la îmbunătățiri semnificative în reducerea ciclurilor de productie.

3.3.2. Analiza răspunsurilor primite și corelarea lor cu ipotezele cercetării

CONFIRMAREA IPOTEZELOR DE CERCETARE PE BAZA INTERVIURILOR

Ipoteza 1. Metodele exacte oferă rezultate superioare celor euristice în programarea producției de serie. Interviurile realizate sugerează un interes variabil în utilizarea algoritmilor IA pentru optimizarea producției, majoritatea companiilor neutilizând încă aceste tehnologii. Cu toate acestea, cei care au început să exploreze utilizarea IA remarcă beneficiile clare în reducerea erorilor și creșterea eficienței. O parte semnificativă a intervievaților subliniază necesitatea unei analize cost-beneficiu detaliate înainte de implementare, ceea ce susține faptul că metodele exacte, deși superioare în rezultate, pot implica costuri semnificative și provocări de integrare. Aceasta sugerează că, în ciuda eficienței metodelor exacte, companiile ezită în adoptarea lor fără o asigurare clară a rentabilității, confirmând astfel ipoteza că metodele exacte oferă rezultate superioare, dar sunt mai greu de implementat fără un angajament ferm din partea managementului și resurse dedicate. Ipoteza este confirmată/

Ipoteza 2. Algoritmii genetici hibrizi oferă rezultate superioare celor genetici clasici în ceea ce privește exactitatea soluțiilor oferite în programarea producției de serie. Din interviuri reiese o percepție generală că algoritmii genetici hibrizi, deși neexplorați pe larg în practică, au potențialul de a oferi soluții mai precise în programarea producției de serie, comparativ cu metodele tradiționale. Totuși, acceptarea acestor tehnologii depinde de modul în care sunt prezentate beneficiile și de pregătirea angajaților pentru a le utiliza eficient. Companiile care au început să testeze sau să se gândească la implementarea acestor tehnologii recunosc necesitatea unui sprijin managerial puternic și a unei colaborări interdepartamentale pentru a asigura succesul acestora. Acest lucru confirmă ipoteza că algoritmii genetici hibrizi pot oferi avantaje considerabile în programarea producției, dar necesită o pregătire riguroasă și o înțelegere aprofundată a tehnologiei. Ipoteza se confirmă.

Ipoteza 3: Algoritmii genetici clasici oferă rezultate diferite și neconcludente în rulările succesive în programarea producției de serie mare. Interviurile indică faptul că majoritatea companiilor încă nu utilizează algoritmi genetici, fie clasici, fie hibrizi, în programarea producției. Printre cei care au experimentat cu tehnologiile IA, există o recunoaștere a limitărilor acestora, în special în privința consistenței rezultatelor. Mai multe companii au subliniat că lipsa unor date concrete și fiabile face ca implementarea acestor algoritmi să fie provocatoare, ceea ce sugerează că rezultatele obținute prin metode genetice clasice pot fi inegale și dificil de replicat. Aceasta confirmă ipoteza că algoritmii genetici clasici nu oferă întotdeauna rezultate concludente, ceea ce poate descuraja utilizarea lor pe scară largă în programarea producției de serie mare. Ipoteza se confirmă.

Ipoteza 4. Aplicarea modelului Guinet în programarea producției va conduce la îmbunătățiri semnificative în reducerea ciclurilor de producție. Din analiza interviurilor, reiese că sprijinul managerial și infrastructura tehnologică sunt factori esențiali pentru succesul

implementării IA, ceea ce ar sugera că modele complexe, precum cel al lui Guinet, ar avea nevoie de un angajament și resurse substanțiale pentru a fi implementate eficient. Companiile au subliniat importanța unei modelări adecvate și a adaptării tehnologiei la nevoile lor specifice pentru a obține o reducere semnificativă a timpilor de producție. Aceasta confirmă ipoteza că aplicarea modelului Guinet poate aduce îmbunătățiri semnificative, dar necesită o planificare și o execuție riguroasă, precum și o infrastructură solidă și un sprijin continuu din partea tuturor părților implicate. Ipoteza este confirmată.

Ipoteza 5. Rezultatele aplicării algoritmilor hibrizi au șanse ca pe viitor să se apropie de rezultatele aplicării metodelor exacte în ceea ce privește programarea producției de serie mare. Interviurile sugerează o percepție pozitivă asupra potențialului algoritmilor hibrizi, deși aceștia nu sunt încă larg adoptați. Multe companii sunt deschise la ideea de a investi în IA, recunoscând că în viitor aceste tehnologii ar putea deveni la fel de eficiente ca metodele exacte. Acceptarea graduală a acestor tehnologii, susținută de investiții și instruire adecvată, ar putea conduce la o creștere a preciziei și eficienței în programarea producției de serie mare. Aceasta susține ipoteza că, pe măsură ce tehnologia avansează și este mai bine înțeleasă, algoritmii hibrizi ar putea să ofere soluții comparabile cu cele ale metodelor exacte, făcându-i astfel o alternativă viabilă în viitorul apropiat. Ipoteza se confirmă.

3.4. Studiu de Caz: ABC Romania

Identificarea unui model de optimizare a procesului de planificare a producției presupune, în mod necesar, o colaborare între cercetător și o societate în care tiparul propus să poată fi implementat, astfel încât dezideratul urmărit prin cercetarea de față să poată fi atins.

Întreprinderile performante sunt cele ale căror factori decizionali știu să se adapteze rapid la noile condiții în schimbare și să identifice oportunitățile și metodele viabile de îmbunătățire a vectorilor aplicați. Prospectiva constă în elaborarea mai multor scenarii probabile ale viitorului, care țin seama de tendințele trecute, considerând ipoteza apariției unor crize sau rupturi în echilibrul activității de producție.

Partenerul nostru în această cercetare este firma ABC din Cluj-Napoca, subsidiară a DEF și membră a grupului XYZ cu o cifră de afaceri de peste 14 milioane euro. Firma deține 8000 mp de hale, produce peste 350 de produse lunar, exportă în 8 țări și investește anual 0,5 milioane euro.

Domenii de activitate:

- Producția de subansamble metalice pentru industria aparaturii electrice casnice.
- Asamblare și testare electromecanică.

Organizatoric, firma are 17 centre de producție, mașinile fiind grupate în centre specifice anumitor operații. Importanța în general a domeniului de studiu a cercetării este crucială pentru optimizarea și eficientizarea întregului proces de producție, contribuind direct la reducerea timpilor de livrare și la creșterea competitivității firmei pe piață.

Unele dintre cele mai mari probleme cu care se confruntă numeroase firme din economia românească se regăsesc în gestionarea producției, dată de unicitatea produsului finit, care necesită o atenție sporită în procesul de producție. Opiniile concertate în urma sondajelor efectuate, studiate in literatura de specialitate au relevat că stilul de management, definit ca fiind haotic, concretizat în modul defectuos în care echipa de conducere a organizat activitatea firmei, este factorul primordial al apariției problemelor în procesele de producție. Multe dintre aceste situații au apărut din cauza faptului că deciziile de management au fost înțelese eronat de către angajați, consecințele materializându-se în scăderea productivității și calității produselor.

În urma unui studiu aprofundat, concretizat prin analiza modului în care se desfășurau procesele de producție, s-a concluzionat că geneza problemelor se regăsea în planificarea și programarea producției. Multe dintre problemele ivite în procesul de planificare a producției au fost identificate în gestionarea fluxului de materii prime. Punctul de plecare al livrării produsului finit îl constituie lansarea comenzii, etapa cuprinsă între acești doi indicatori tempo-spațiali necesitând un traseu clar, predictibil și previzibil al materiilor prime.

Au fost, astfel, identificați în literatura de specialitate ca factori perturbatori ai activității de producție lipsa de experiență a unor angajați, ceea ce a condus la transmiterea materiilor prime către alte zone de lucru decât cele cărora le erau destinate. Consecința directă a fost că produsele din comandă nu erau executate în conformitate cu așteptările și standardele clientului. Divagarea de la specificații era constatată abia la etapa următoare de prelucrare a produsului, iar remedierea se putea efectua doar prin retrimiterea acestuia în etapa anterioară de producție, deja parcursă o dată, aspect care dubla fluxul de producție și provoca întârzieri în livrarea produsului finit către client.

Un alt factor perturbator în respectarea planului și a termenului de livrare a fost depistat în stocarea materiilor prime și a materialelor, atât în incinta halei de producție, cât și în afara acesteia, care aveau un impact negativ asupra operațiilor de descărcare a camioanelor cu materii prime și materiale.

Încălcarea clauzelor privitoare la termenele de livrare a comenzilor, un alt element ce afectează procesul de predare a produsului finit, era cauza directă a adoptării unor valori nerealiste, net inferioare pentru normele de timp, acestea fiind determinate fără a lua în considerare capacitatea de funcționare reală a utilajelor ori a forței de muncă. În concluzie, s-a constatat că gradul de utilizare a capacității productive s-a limitat la o valoare inferioară față de ceea ce, în mod real, s-ar fi putut realiza.

O nouă metodă de eficientizare a procesului de producție a fost identificată ca fiind mentenanța preventivă a utilajelor, operațiune ignorată, în mod paradoxal, cu scopul de a se asigura

respectarea termenelor de livrare. Aceasta deoarece defecțiunile ivite la echipamentele de producție erau aduse la cunoștința persoanelor competente mult prea târziu, situație care determina blocarea fluxului productiv pentru ore sau chiar zile, în funcție de gravitatea defecțiunilor echipamentelor.

Controlul tehnic de calitate s-a dovedit necesar la finalul fiecăreia dintre etapele de producție, astfel încât eventualele erori de execuție să poată fi identificate la timp și nu în etape ulterioare, care ar impune retrimiterea produsului spre remediere într-o etapă anterioară, deja efectuată, pentru a asigura dezideratul calității produselor finite.

Nu este lipsită de importanță nici programarea activităților de remediere a erorilor de execuție. Atunci când societatea nu dispune de suport informatic, programarea acestor activități se efectuează anevoios, iar comanda era livrată clientului cu depășirea termenelor stabilite.

În elaborarea metodelor optime pentru îmbunătățirea parametrilor de producție, Guinet și colegii săi au conceput un model care ia în considerare grupuri de mașini capabile să proceseze comenzi în paralel. Astfel, a fost definită noțiunea de stadiu (etapă, fază), care oferă posibilitatea realizării operațiilor aferente unei etape pe oricare dintre echipamentele alocate stadiului respectiv.

Elocvența modelului propus constă în capacitatea sa de a permite determinarea ordinii în care trebuie procesate comenzile și momentul temporal în care se finalizează procesarea fiecărei comenzi pe fiecare set de mașini.

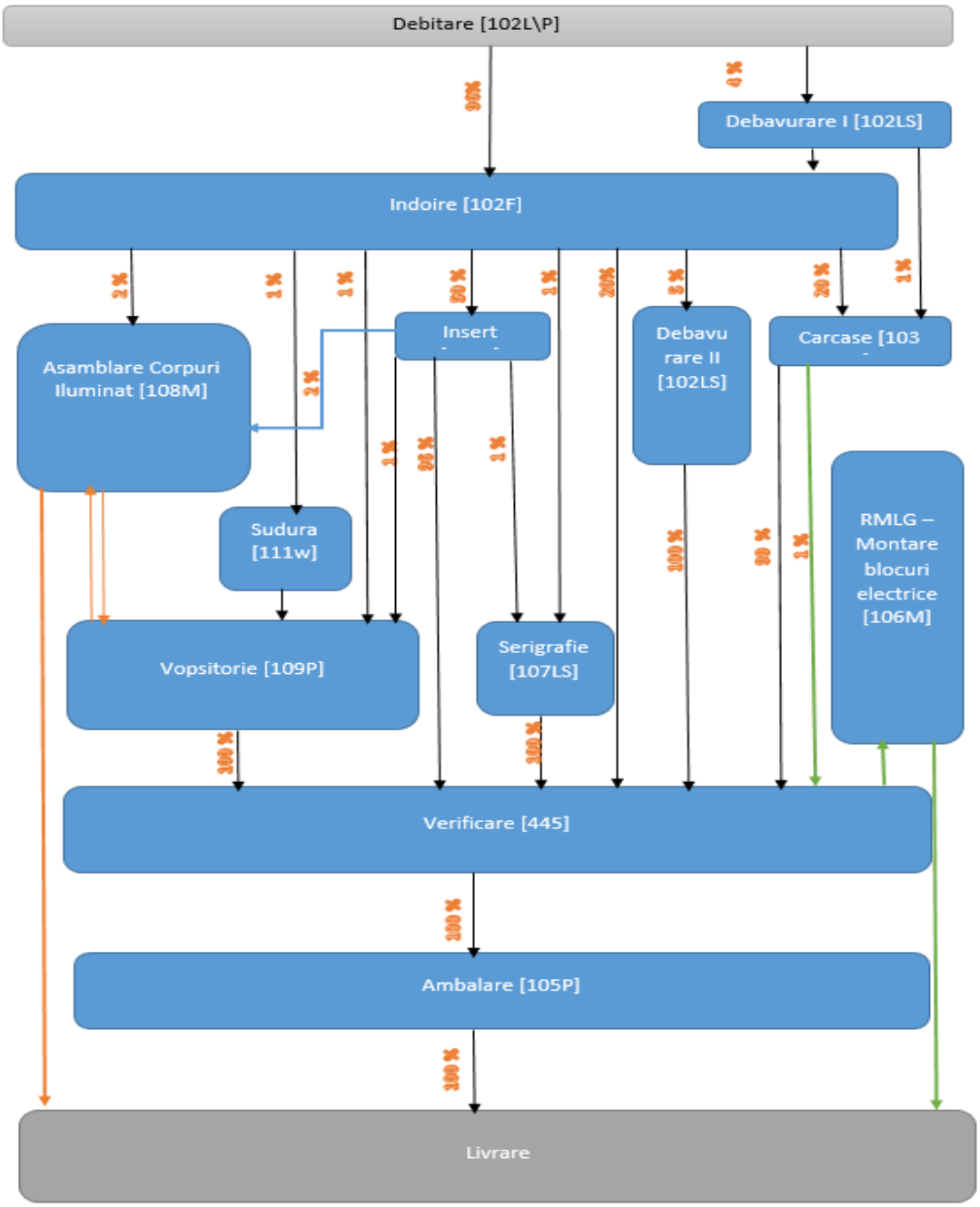


Figura 3.2. Schema Tehnologică ABC
(sursa: ABC Romania)

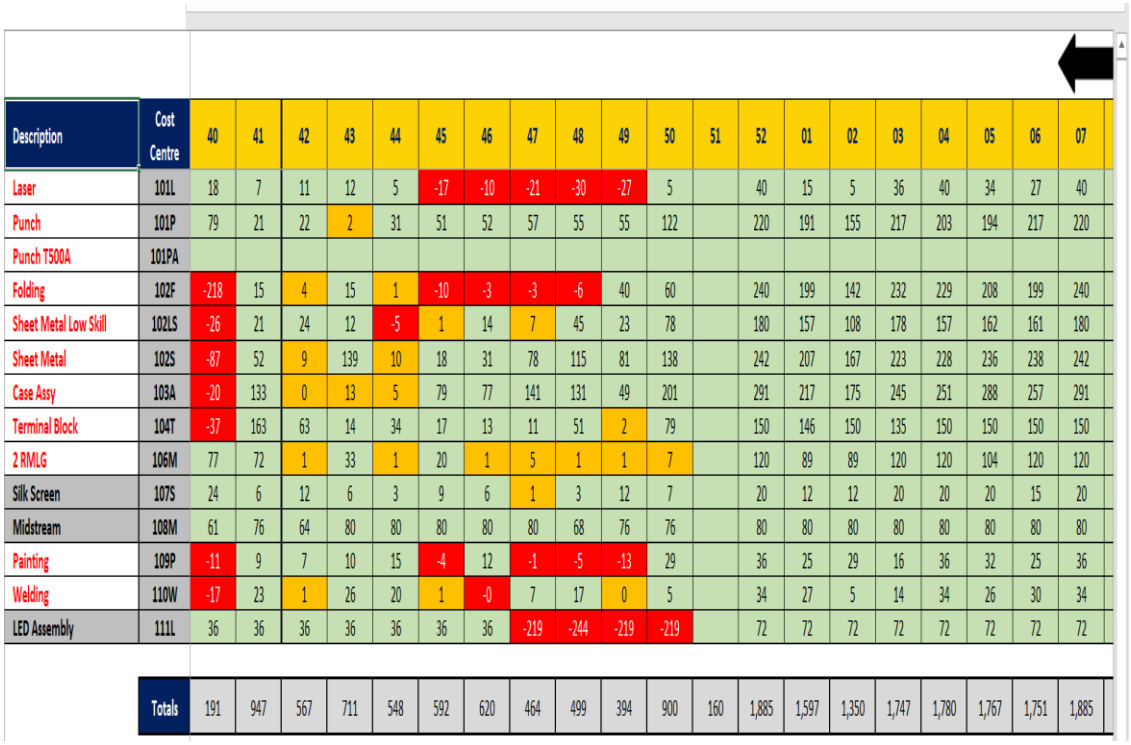
Schema tehnologică a liniilor de fabricație. Prima etapă a procesului de producție este debitarea, executată în 2 centre de producție (102 L/P): unul cu o mașină de debitare cu laser și celălalt cu 6 mașini de ștanțare. O parte din piese sunt trimise apoi la debavurare (102LS), iar restul trec direct la îndoire (102 F). Urmează ca fiecare produs să treacă prin centrele de producție specifice itinerarului tehnologic dedicat: Inserturi, Realizare Carcase, Asamblare Corpuri Iluminat, Sudură, Serigrafie, Vopsitorie. La final, produsele trec prin verificare (centrul de producție 445 D) și, în funcție de

solicitările clienților, prin Montare Blocuri Electrice (106M). Înainte de livrare, produsele sunt ambalate (105P). Fiecare centru de producție conține între 1 și 10 mașini specializate.

Îmbunătățirea programării producției. Procesele tehnologice din cadrul firmei ABC necesită îmbunătățirea programării producției, așa cum este descris în tabelul următor.

Utilizarea sistemului Pegasus duce la situații în care în centrele de producție echipate cu mașini de debitare și îndoire se evidențiază apariția ”locurilor înguste”, care reprezintă puncte critice în fluxul de producție(a se vedea în Tabelul Nr 3.1.). Aceste locuri înguste limitează capacitatea centrelor de producție de a finaliza toate comenzile programate în intervalul de timp prevăzut, conducând la întârzieri și la o eficiență operațională redusă. Această problemă subliniază necesitatea optimizării fluxurilor de producție și a gestionării adecvate a resurselor pentru a minimiza impactul acestor constrângeri și pentru a asigura respectarea termenelor de livrare fără a crește semnificativ impactul asupra costurilor.

Tabel 3.1. Exemplificare întârzieri în săptămâna 40 anul 2021



Description	Cost Centre	40	41	42	43	44	45	46	47	48	49	50	51	52	01	02	03	04	05	06	07
Laser	101L	18	7	11	12	5	-17	-10	-21	-30	-27	5		40	15	5	36	40	34	27	40
Punch	101P	79	21	22	2	31	51	52	57	55	55	122		220	191	155	217	203	194	217	220
Punch T500A	101PA																				
Folding	102F	-218	15	4	15	1	-10	-3	-3	-6	40	60		240	199	142	232	229	208	199	240
Sheet Metal Low Skill	102LS	-26	21	24	12	-5	1	14	7	45	23	78		180	157	108	178	157	162	161	180
Sheet Metal	102S	-87	52	9	139	10	18	31	78	115	81	138		242	207	167	223	228	236	238	242
Case Assy	103A	-20	133	0	13	5	79	77	141	131	49	201		291	217	175	245	251	288	257	291
Terminal Block	104T	-37	163	63	14	34	17	13	11	51	2	79		150	146	150	135	150	150	150	150
2 RMLG	106M	77	72	1	33	1	20	1	5	1	1	7		120	89	89	120	120	104	120	120
Silk Screen	107S	24	6	12	6	3	9	6	1	3	12	7		20	12	12	20	20	20	15	20
Midstream	108M	61	76	64	80	80	80	80	80	68	76	76		80	80	80	80	80	80	80	80
Painting	109P	-11	9	7	10	15	-4	12	-1	-5	-13	29		36	25	29	16	36	32	25	36
Welding	110W	-17	23	1	26	20	1	-0	7	17	0	5		34	27	5	14	34	26	30	34
LED Assembly	111L	36	36	36	36	36	36	36	-219	-244	-219	-219		72	72	72	72	72	72	72	72
	Totals	191	947	567	711	548	592	620	464	499	394	900	160	1,885	1,597	1,350	1,747	1,780	1,767	1,751	1,885

(sursa: Programul Pegasus al ABC)

Formule de optimizare a ciclurilor de producție. Pentru a aborda problema sistemic și etapizat, vom defini o funcție obiectiv al cărei s cop este minimizarea ciclurilor de producție. Analiza rezultatelor obținute, comparate cu situația existentă, va duce la concluzii care vor sta la baza modificării ordinii de intrare în producție a loturilor.

Setul de date inițiale utilizat pentru analiza de față include o perioadă de programare a producției de 3 săptămâni, corespunzătoare unui număr de 280 de comenzi pentru 353 de produse fabricate de compania ABC. Fiecare dintre aceste produse este asociat unui itinerar tehnologic specific, detaliat în figura următoare.

Cod Produ	Timp total de prod	M001	M02	M02	M031	M036	M03	M038	M040	M042	M043	M04	M048	M049	M
ABC 001	0.37422	0.05422								0.32					
ABC 002	6.325	2.425							2.4	1.5					
ABC 003	6.325	2.425							2.4	1.5					
ABC 004	0.32									0.32					
ABC 005	0.32									0.32					
ABC 006	8.255	3.355							3.4	1.5					
ABC 007	7.405	3.555							2.4	1.45					
ABC 008	0														
ABC 009	14.05		3.5		0.5				4.7					3.55	
ABC 010	15.45		3.5		1.9				4.7					3.55	
ABC 011	0														
ABC 012	13.75		3.5		0.5				4.7					3.25	
ABC 013	0														
ABC 014	13.85		3.5		0.5				4.7					3.35	
ABC 015	15.25		3.5		1.9				4.7					3.35	
ABC 016	13.55		3.5		0.5				4.7					3.05	
ABC 017	5.55		0.6								0.4		3.5		
ABC 018	0.63104		0.21												
ABC 019	0.59673		0.21												
ABC 020	1.13075	0.33075								0.8					
ABC 021	0.28175	0.28175													
ABC 022	5.58		0.4								0.4		3.73		
ABC 023	0.25817	0.05617								0.202					
ABC 024	0.2				0.2										
ABC 025	0.33261		0.04				0.1			0.06					
ABC 026	0.33261		0.04				0.1			0.06					
ABC 027	5.48211	0.28211	1.5		0.4		0.25								
ABC 028	0.51683	0.16983								0.347					
ABC 029	13.55		3.5		0.5				4.7					3.05	
ABC 030	0.32									0.32					
ABC 031	7.2988	2.5988							3	1.7					
ABC 032	6.84605	2.34605							3	1.5					
ABC 033	1.00196				0.25	0	0.2							0.125	
ABC 034	0.96189	0.27489			0.25					0.202					
ABC 035	0.58665	0.11665			0.2					0.2					
ABC 036	2.05905	0.22905			0.13		0.35			0.7					
ABC 037	2.54	0.41	0.65		0.35						0.8				
ABC 038	1.846		0.4		0.25						0.8				
ABC 039	1.55248	0.56048			0.25										
ABC 040	1.947	0.147	0.3		0.2		0.5				0.8				

Figura 3.3. Captură de ecran privind exemplu de itinerar tehnologic al produselor fimei ABC (sursa: prelucrare proprie)

Itinerarele tehnologice reprezintă succesiunile precise de operații și mașini necesare pentru fabricarea fiecărui produs, reflectând complexitatea și diversitatea proceselor de producție din cadrul firmei.

În contextul celor 280 de comenzi, fiecare comandă necesită o planificare atentă pentru a maximiza utilizarea eficientă a resurselor și pentru a respecta termenele de livrare stabilite.

Fiecare produs din setul de date nu doar că are un itinerar tehnologic unic, dar aceste itinerare pot varia considerabil în ceea ce privește complexitatea, timpul necesar și ordinea operațiilor. Aceasta introduce un nivel suplimentar de dificultate în optimizarea programării, deoarece trebuie să se asigure o distribuție echilibrată a sarcinilor între mașini și echipamente, evitând astfel suprasolicitarea unora și subutilizarea altora.

Documentarea detaliată a acestor itinerare tehnologice din figura anterioară oferă o imagine clară asupra cerințelor de producție și servește drept fundament pentru algoritmii de programare

utilizați în acest studiu. Astfel, acest set de date reprezintă un exemplu complex, dar tipic, de provocare în programarea producției, unde soluțiile eficiente sunt esențiale pentru succesul operațional al firmei ABC.

Demersul cercetării noastre îl reprezintă un răspuns direct la problema locurilor înguste identificate în cadrul centrelor de producție, cu scopul de a elimina aceste blocaje și de a optimiza fluxul de producție. În urma unei analize detaliate a situației existente, am constatat că aplicarea modelului matematic propus de Guinet poate oferi o soluție eficientă pentru rezolvarea acestei provocări. Modelul lui Guinet, recunoscut pentru capacitatea sa de a optimiza alocarea resurselor și de a reduce timpii de așteptare în procesele de producție, a fost particularizat pentru nevoile specifice ale companiei ABC. Această particularizare s-a realizat prin adaptarea parametrilor modelului în conformitate cu datele inițiale prezentate mai sus, privind fluxurile de producție, itinerariile tehnologice și constrângerile operaționale existente. Astfel, prin integrarea acestor date în modelul matematic, cercetarea noastră nu doar că propune o soluție teoretică pentru eliminarea locurilor înguste, dar și oferă o aplicabilitate practică ce poate fi implementată direct în mediul de producție al companiei ABC, dar și a altor companii care folosesc modele de produție similare Acest demers contribuie la creșterea eficienței operaționale și la asigurarea unui flux continuu și optimizat al producției, reducând astfel riscul de întârzieri și îmbunătățind capacitatea companiei de a răspunde cerințelor pieței.

Aplicarea modelului Guinet în cadrul procesului de producție a condus la eliminarea locurilor înguste, optimizând astfel fluxul de producție. Această intervenție a permis ordonanțarea eficientă a producției, asigurând un flux continuu și reducând la minimum timpii de așteptare. Ca rezultat, s-a reușit eliminarea întârzierilor în livrarea produselor, îmbunătățind considerabil performanța operațională și capacitatea de a respecta termenele de livrare.

Ca atare, am ajuns la următoarea funcție obiectiv:

$$f = min \sum_{i=1}^{I} T_{i,M_p} \quad (1)$$

Constrângerile aferente acestei funcții obiectiv sunt :

$$T_{i,m_p} - T_{j,m_p} + KZ_{i,j,m_p} \geq d_{j,m_p} \quad \forall i,j \in [1,I] \ i \neq j \ \forall m_p \in [1,M_P] \quad (2)$$

$$T_{j,M_p} - T_{i,M_p} + K(1 - Z_{i,j,m_p}) \geq d_{i,m_p} \quad \forall i,j \in [1,I] \ i \neq j \ \forall m_p \in [1,M_P] \quad (3)$$

$$T_{i,m_p+1} - T_{i,m_p} \geq d_{i,m_p} \quad \forall i, \in [1,I] \quad \forall m_p \in [1,M_P - 1] \quad (4)$$

$$T_{i,1} \geq dint_i \quad \forall i \in [1, I] \tag{5}$$

$$T_{i,M_P} \leq dl_i - d_{i,M_P} \; \forall \; i \in [1, I] \tag{6}$$

unde:

$T_{i,Mp}$ − Timpul necesar operaţiei i, pe centru de producţie m şi maşină p

$T_{j,Mp}$ − Timpul necesar operaţiei j, pe centru de producţie m şi maşina p

$Z_{i,j,m}$ − Variabila cu val 1 când operaţia i e programată înaintea operaţiei j pe centrul m

$Z_{i,j,m}$ − Variabila cu val 0 când operaţia j e programată înaintea operaţiei i pe centrul m

$d_{i,Mp}$ − Timpul de procesarea operaţiei i pe centru de producţie m şi maşina p

$d_{j,Mp}$ − Timpul de procesarea operaţiei j pe centru de producţie m şi maşina p

$dint_i$ − Momentul de la care poate fi programată comanda i

dl_i − Momentul la care comanda i trebuie livrată

K − constanta ce are o valoare mare

M − Numărul centrelor de producţie

I, J − numărul de comenzi

P − numărul de maşini din fiecare centru de producţie

Cu limitele valorice:

i,j ∈ [1 , 5000]

m ∈ [1 , 17]

P ∈ [1 , 54]

p ∈ [1 , 353]

Constrângerile (2) și (3) împiedică suprapunerea în timp a două comenzi i si j programate în același centru de producție m, pe aceiași mașina. Constrângerile (4) joacă un rol similar, doar că, în acest caz, sunt evitate suprapunerile prelucrării comenzii i pe mașinile m+1 si m. Constrângerile (5) si (6) asigură atribuirea de valori pentru variabilele $T_{i,mp}$, astfel încât prelucrările comenzilor să nu înceapă mai repede decât momentul de intrare în evidența firmei, dar nici mai târziu decât termenul de livrare.

3.4.1. Integrarea și implementarea tehnologiilor

Compania ABC utilizează în acest moment facilitățile oferite de modelul Pegasus care integrează o serie de activități. Este folosit de asemenea si modulul de programare a producției oferit de această companie.

Modelul **Pegasus** pentru planificarea comenzilor într-o companie de producție este un sistem robust, conceput pentru a eficientiza procesele de fabricație. Integrează diverse funcționalități esențiale pentru îmbunătățirea planificării producției, gestionarea comenzilor și alocarea resurselor. În continuare, vom analiza principalele caracteristici ale modelului Pegasus și importanța lor în contextul unei companii de producție. [114]

Planificarea producției este un aspect critic pentru succesul operațiunilor de fabricație. Pegasus oferă un instrument cuprinzător de planificare care ajută la crearea programelor detaliate de producție. Acest proces include organizarea procesului de fabricație, urmărirea progresului și ajustarea programelor în funcție de schimbările în cerere sau capacitate de producție. Lockamy și Khurana (1995) subliniază că o planificare eficientă poate reduce semnificativ timpii de livrare și costurile de producție. În plus, planificarea producției ajută la evitarea întreruperilor și a întârzierilor, asigurându-se că producția răspunde eficient cerințelor pieței. [114]

Procesarea comenzilor de vânzări. Modulul de procesare a comenzilor de vânzări din Pegasus permite gestionarea eficientă a comenzilor, de la momentul intrării până la livrare. Acesta ajută la crearea unei liste de materii prime necesare pentru producție, estimarea costurilor și gestionarea programelor de livrare. Un sistem eficient de procesare a comenzilor contribuie la reducerea erorilor și la îmbunătățirea relațiilor cu clienții. Astfel, fiecare comandă este procesată cu acuratețe și livrată la timp, îmbunătățind satisfacția clienților și consolidând reputația companiei. [115]

Managementul stocului și al resurselor. Pegasus include instrumente pentru gestionarea stocului, asigurându-se că materiile prime și componentele sunt disponibile atunci când este necesar. Gestionarea eficientă a stocului este crucială pentru reducerea costurilor operaționale și îmbunătățirea fluxului de numerar. De exemplu, gestionarea adecvată a stocului poate reduce costurile de depozitare și riscul de perimare a stocurilor.

Programul principal de producție (en. Master Production Schedule- MPS). MPS-ul din Pegasus ajută la adaptarea capacității de producție cu cererea pieței. Acesta detaliază ce produse vor fi fabricate, când și în ce cantități. Un MPS bine conceput poate preveni lipsurile de stoc, controla costurile și îmbunătăți eficiența generală a producției. Stevenson (2005) subliniază că un MPS eficient este esențial pentru menținerea unui echilibru între oferta și cerere, minimizând astfel pierderile și maximizând profitabilitatea. [116]

Trasabilitatea. Modulul de trasabilitate din Pegasus permite urmărirea materialelor și componentelor de la furnizori până la livrare. Trasabilitatea este esențială pentru controlul calității și în cazul rechemărilor de produse, deoarece permite identificarea loturilor specifice și a utilizării acestora în producție. Trasabilitatea joacă un rol vital în menținerea standardelor de calitate și în reacționarea rapidă la problemele de calitate.

Modelul Pegasus pentru programarea comenzilor într-o companie de producție reprezintă un instrument valoros pentru optimizarea proceselor de fabricație. Prin integrarea planificării producției, gestionării comenzilor de vânzări, gestionarea procesului de inventariere, programului principal de producție și trasabilității, Pegasus ajută companiile să răspundă eficient cerințelor pieței, să controleze costurile și să mențină standardele de calitate. Aceste beneficii contribuie la creșterea competitivității și a performanței generale a companiei pe piață

Modulul de programare a producției din cadrul modelului Pegasus este conceput pentru a optimiza procesul de fabricație prin integrarea planificării detaliate, monitorizării în timp real și ajustărilor adaptive pentru a asigura eficiența și a satisface cerințele pieței. În continuare va fi analizat modul cum funcționează în detaliu:

Prin utilizarea modelului Pegasus, companiile pot atinge o eficiență operațională ridicată, asigurând livrarea la timp a produselor și menținerea calității.

1. Programarea procesului de producție

a) Programarea inițială cuprinde:

- **Definirea comenzilor de producție** începe cu crearea comenzilor de producție bazate pe previziunile contractuale, comenzile existente și nivelurile de stoc. Acesta include specificarea cantităților ce urmează a fi produse, termenele limită și secvența operațiunilor. De exemplu, într-o companie de producție auto, comenzile de producție sunt planificate pentru a răspunde cererii sezoniere și lansărilor de noi modele.
- **Alocarea resurselor** precum materiile prime, forța de muncă și mașinile. Acest lucru asigură că toate componentele necesare sunt disponibile și că există suficiente ore de muncă și ore de funcționare a

mașinilor pentru a îndeplini obiectivele de producție. Literatura de specialitate subliniază că o alocare eficientă a resurselor poate reduce timpii morți și poate îmbunătăți productivitatea. [117]

b) Bill of materials (en. BOM)

- **Lista de materiale** - sistemul utilizează BOM pentru a lista toate materialele și componentele necesare pentru producție. Acest lucru ajută la planificarea achizițiilor și gestionarea stocului. În industria electronică, de exemplu, BOM include componente precum circuite integrate, rezistențe și carcase.

2. Crearea programelor detaliate

a) Master Production Schedule (en. MPS)

- **Planificare agregată** - MPS este utilizat pentru a crea o vedere de ansamblu a producției pe o perioadă specifică, de obicei aliniată cu cererea pieței și capacitatea de producție. Aceasta implică echilibrarea cererii prognozate cu capacitățile de producție. Un exemplu de succes este Toyota, care utilizează MPS pentru a coordona producția în fabricile sale globale.

b) Diagrame Gantt și Metoda Kanbann

- **Instrumente de vizualizare** - Programele detaliate sunt vizualizate folosind instrumente precum diagramele Gantt și panoul și cardurile Kanban. Aceste instrumente ajută la programarea fiecărei faze de producție și la vizualizarea fluxului de lucru, facilitând gestionarea proceselor de producție complexe. De exemplu, în industria software, aceste instrumente sunt esențiale pentru gestionarea dezvoltării agile.

3. Urmărirea progresului - monitorizare în timp real

- **Urmărirea progresului**: Colectarea de date în timp real de pe linia de producție permite urmărirea fiecărei etape de producție. Acest lucru include monitorizarea stadiului comenzilor de lucru, utilizarea mașinilor și productivitatea muncii. Key Performance Indicators (KPIs) precum timpul de ciclu, productivitatea și perioadele de inactivitate sunt monitorizate pentru a evalua eficiența și a identifica blocajele.

4. Ajustarea Programelor pe Baza Schimbărilor

a) Fluctuațiile cererii

- **Programare adaptivă** - Sistemul poate să se adapteze schimbărilor în cerere prin ajustarea dinamică a programelor de producție. Dacă există o creștere a volumului comenzilor, programul poate fi

modificat pentru a intensifica producția. Invers, dacă cererea scade, sistemul poate reduce producția pentru a preveni supra-producția.

b) Constrângerile privind capacitatea

- **Re-alocarea resurselor** dacă o mașină se defectează sau există o lipsă de personal. Acest lucru include reprogramarea sarcinilor și redistribuirea sarcinilor către alte resurse disponibile.

c) Planificarea scenariilor

- **Analiza prezumtivă** - sistemul permite planificarea scenariilor, unde pot fi simulate diferite scenarii de programare pentru a înțelege impactul schimbărilor în parametrii de producție. Acest lucru ajută la luarea unor decizii informate pentru optimizarea producției.

5. Asigurarea unei producții eficiente

a) Managementul stocului

- **Producția Just-In-Time (JIT)**: Prin sincronizarea programelor de producție cu nivelurile de stoc, sistemul sprijină producția JIT, reducând costurile de păstrare a stocului și minimizând risipa. Un exemplu este industria auto, unde JIT a fost adoptat pentru a reduce stocurile și a îmbunătăți eficiența.

b) Îmbunătățirea continuă

- **Circuit de feedback**: Datele colectate din procesele de producție sunt analizate pentru a identifica zonele de îmbunătățire. Acest circuit de feedback ajută la rafinarea programelor și îmbunătățirea eficienței generale în timp.

6. Integrarea cu software-ul de management al proiectelor .

- **Funcționalitate îmbunătățită**: Prin integrarea cu software-ul de management al proiectelor precum ProjectManager, modelul Pegasus își îmbunătățește capacitățile. Aceasta include caracteristici precum diagrame Gantt interactive, colaborare în timp real și instrumente avansate de gestionare a resurselor.

Modelul Pegasus pentru programarea producției este un sistem potrivit în optimizarea fluxurilor de lucru în fabrici. Procesul începe cu o comandă de la client, care declanșează crearea unei comenzi de producție în Pegasus. Acest prim pas este esențial pentru a asigura că toate cerințele clientului sunt integrate în planul de producție.

După crearea comenzii de producție, se utilizează BOM și informațiile traseului tehnologic pentru a detalia toate materialele necesare și pentru a contura procesul de producție. BOM este un document esențial care listează toate componentele și materialele necesare pentru fabricarea produsului, asigurându-se astfel că toate resursele sunt disponibile la momentul potrivit.

Crearea programului de producție detaliat este următorul pas important. Folosind diagrame Gantt, Pegasus permite vizualizarea și programare a fiecărei faze a producției. Aceste instrumente de vizualizare sunt esențiale pentru gestionarea fluxurilor de lucru complexe și pentru asigurarea că toate etapele producției sunt sincronizate corect.

Execuția programului de producție implică *a) monitorizarea în timp real a progresului* și *b) ajustările necesare pentru a menține fluxul de producție*. Aceasta permite identificarea rapidă a oricăror întârzieri sau probleme și luarea de măsuri corective. În final, după finalizarea producției, produsele sunt livrate, iar datele colectate sunt analizate pentru îmbunătățiri viitoare. Acest proces continuu de monitorizare și feedback este esențial pentru optimizarea continuă a producției.

Puncte slabe ale software-ului Pegasus sunt [118]:

- Deși modelul Pegasus este robust și benefic pentru un număr mare de comenzi de producție, există anumite slăbiciuni care pot afecta eficacitatea sa generală.
- Unul dintre principalele dezavantaje ale modelului Pegasus este complexitatea sa. Natura cuprinzătoare a acestui sistem poate face implementarea și utilizarea sa dificile, în special pentru afacerile mici sau pentru cele fără expertiză tehnică extinsă. Utilizarea eficientă a modelului Pegasus necesită o instruire semnificativă, ceea ce poate fi atât costisitor, cât și consumator de timp .
- Deși modelul Pegasus oferă multe caracteristici standard, poate să nu integreze ușor procesele de afaceri extrem de specifice sau unice fără o personalizare semnificativă. Acest lucru poate necesita resurse suplimentare și poate fi un proces provocator pentru companiile care au nevoi foarte specifice .
- Integrarea modelului Pegasus cu sistemele existente, cum ar fi ERP sau alte instrumente de management al producției, poate fi problematică. Discrepanțele dintre diferite sisteme pot duce la inconsistențe de date și întreruperi ale fluxului de lucru .
- În timp ce Pegasus poate fi scalat într-o anumită măsură, întreprinderile foarte mari cu operațiuni extrem de complexe ar putea găsi sistemul mai puțin scalabil în comparație cu soluțiile specializate la nivel de întreprindere .
- Costul inițial de implementare al modelului Pegasus poate fi ridicat, mai ales atunci când se iau în considerare cheltuielile legate de instruire, personalizare și integrare. De asemenea, costurile de întreținere și suport pot duce la creșteri ale costului total, în special dacă sistemul necesită actualizări frecvente sau modificări .

- Eficacitatea modelului Pegasus depinde în mare măsură de acuratețea și actualitatea datelor introduse. Datele inexacte pot duce la decizii de programare eronată a producției și la ineficiență în producție .
- Deși modelul sprijină programarea adaptivă, s-ar putea să nu fie la fel de receptiv la schimbările bruște și semnificative în cerere sau la întreruperile lanțului de aprovizionare în comparație cu sistemele mai agile .
- Angajații obișnuiți cu sisteme mai vechi sau mai simple pot opune rezistență adoptării modelului Pegasus din cauza complexității sale și a percepției că este dificil de învățat un nou sistem.

În concluzie, modelul Pegasus reprezintă un instrument bun pentru programarea și gestionarea producției, dar succesul său depinde de capacitatea unei companii de a gestiona complexitatea, costurile și integrarea acestuia cu alte sisteme. Implementarea și utilizarea eficientă necesită o planificare atentă și resurse adecvate pentru a depăși provocările inerente.

Deși modelul Pegasus oferă un set complet de instrumente pentru programarea producției, punctele slabe mai sus menționate subliniază importanța evaluării adaptabilității sale pentru nevoile specifice ale afacerii și pregătirii adecvate pentru provocările de implementare și integrare. Companiile trebuie să cântărească acești factori în raport cu beneficiile modelului pentru a lua o decizie informată.

Tabel 3.2. Exemplu de programare a producției folosind Pegasus

Part Code ABC	Qty	Start Date	WO Due Date	Start Date Day	WO Due

					Date Day
ABC 001	236	24.05.2023	31.05.2023	143	150
ABC 002	162	31.05.2023	07.06.2023	150	157
ABC 003	108	26.05.2023	02.06.2023	145	152
ABC 003	162	14.06.2023	21.06.2023	164	171
ABC 004	144	31.05.2023	07.06.2023	150	157
ABC 004	144	14.06.2023	21.06.2023	164	171
ABC 005	128	26.05.2023	02.06.2023	145	152
ABC 005	160	14.06.2023	21.06.2023	164	171
ABC 006	144	31.05.2023	07.06.2023	150	157
ABC 006	144	14.06.2023	21.06.2023	164	171
ABC 007	128	26.05.2023	02.06.2023	145	152
ABC 007	160	14.06.2023	21.06.2023	164	171
ABC 008	28	06.06.2023	09.06.2023	156	159
ABC 008	56	13.06.2023	16.06.2023	163	166
ABC 008	28	27.06.2023	30.06.2023	177	180
ABC 008	28	23.05.2023	26.05.2023	142	145
ABC 008	28	23.05.2023	26.05.2023	142	145
ABC 009	28	17.05.2023	24.05.2023	136	143
ABC 009	28	17.05.2023	24.05.2023	136	143
ABC 009	28	17.05.2023	24.05.2023	136	143

(sursa: ABC Romania)

În tabelul anterior poate fi găsit un plan de producție din software-ul Pegasus, unde principalele coloane din tabel sunt:

1. Part Code: Identificarea (codul) piesei produse.
2. ABC: Codul adaptat al piesei
3. Qty: Cantitatea de piese care trebuie produse.
4. Start Date: Data la care este presupus a fi programată să înceapă producția.
5. WO Due Date: Data la care ordinul de lucru trebuie să fie finalizat.

6. Start Hour: Ora zilei la care începe producția.
7. WO Due Hour: Ora zilei la care ordinul de lucru trebuie finalizat.
8. Current CC: Centrul de cost actual, unde are loc producția.
9. Current MC: Centrul mașinii actuale, indicând ce mașină este utilizată.
10. Job No.: Numărul comenzii de producție.
11. SON: Numărul comenzii de vânzare.
12. Customer: Indică dacă producția este pentru stoc sau pentru un client specific.
13. Value: Valoarea financiară a comenzii.

Explicație detaliată a coloanelor cheie:

- **Start Date și Start Hour**: Aceste coloane specifică momentul exact de începere a producției, asigurând că programarea este precisă până la oră.
- **WO Due Date și WO Due Hour**: Similar cu timpul de începere, aceste coloane furnizează termenul limită pentru finalizarea ordinului de lucru, permițând programarea și urmărirea detaliată.
- **Current CC și Current MC**: Aceste coloane urmăresc locul unde se desfășoară producția în cadrul facilității, atât în termeni de centre de cost, cât și de mașini specifice. Acest lucru este crucial pentru alocarea resurselor și monitorizarea utilizării mașinilor.
- **Job No.** Numarul comenzii și numerele comenzilor de vânzare sunt folosite pentru a urmări și gestiona comenzile de producție, legându-le de comenzi specifice ale clienților sau de activitățile de completare a stocului.
- **Customer și Value**: Indică dacă producția este pentru stoc sau pentru un client specific, împreună cu valoarea financiară asociată. Acest lucru ajută la prioritizarea comenzilor și gestionarea costurilor stocului.

Îmbunătățirea planificării producției [119] utilizând aceste date cuprinde:

1. **Acuratețea datelor**. Menținerea înregistrărilor din coloanele relevante actualizate și precise este crucială pentru luarea deciziilor informate și evitarea erorilor de programare. Datele corecte contribuie la prevenirea întârzierilor și optimizează utilizarea resurselor.
2. **Actualizări în timp real**. Implementarea unui sistem care să permită actualizări în timp real ale câmpurilor relevante asigură reflectarea exactă a stării actuale a producției, a întârzierilor și a schimbărilor. Monitorizarea în timp real permite reacții rapide la orice probleme, reducând astfel impactul negativ asupra fluxului de producție.

3. **Prioritizare**. Utilizarea eficientă a câmpului de prioritate ajută la gestionarea fluxului de producție, asigurând că comenzile urgente sunt tratate cu promptitudine. O prioritizare corectă îmbunătățește timpul de răspuns și satisface cerințele clienților în mod eficient.
4. **Alocarea resurselor**. Optimizarea alocării resurselor utilizând datele despre centrele de cost și centrele de mașini ajută la evitarea blocajelor. Aceasta contribuie la utilizarea eficientă a capacităților de producție și la reducerea timpilor de inactivitate.
5. **Monitorizare și ajustări**. Monitorizarea regulată a programului și ajustările efectuate pe baza datelor în timp real permit rezolvarea promptă a oricăror probleme. Adaptarea continuă a programului de producție, bazată pe monitorizarea constantă, ajută la menținerea eficienței și la atingerea obiectivelor de producție.

Acest plan de producție oferă o metodă structurată pentru gestionarea și programarea activităților de producție, asigurând atingerea eficientă și la timp a obiectivelor de producție.

Pentru a îmbunătăți acest tip de programare, conducătorul unității a implementat și metoda Metoda Magee și Boodman. Această abordare suplimentară este menită să optimizeze procesele și să sporească eficiența generală a programului, aducând beneficii semnificative în gestionarea resurselor și în obținerea rezultatelor dorite.

Metoda Magee și Boodman selectează din setul dat subseturi de comenzi care să asigure pe intervale de timp, de exemplu pe zile, o încărcare echilibrată a tuturor maşinilor de pe itinerarul tehnologic. [63]

Astfel în primul rând, ținând cont de numărul de schimburi pe zi, se dorește aflarea unui grup de comenzi care să asigure fiecărei maşini un volum de prelucrări cât mai apropiat de maximum posibil de prelucrare a mașinii în funcție de numarul de schimburi zilnice. Astfel comenzile care sunt primite se grupează pe zile astfel încât să nu se depășească, numărul maxim de ore de prelucrare a mașinilor dar această valoare să se apropie cat mai mult de limita sa superioară. În urma acestui proces se genereaza o listă de progrmare a producției pe fiecare zi.

Această abordare euristică se bazează pe ideea că, în loc să caute o soluție perfectă sau optimă în sens matematic, se dezvoltă o soluție „suficient de bună” care poate fi aplicată în condiții reale de producție. Acest lucru este esențial, având în vedere complexitatea și variabilitatea proceselor industriale. De asemenea, metoda are avantajul de a fi relativ simplu de implementat în comparație cu alte metode mai sofisticate, cum ar fi optimizarea liniară sau metodele exacte de programare matematică. În plus, procedura se adaptează specificităților fiecărei comenzi și ale fiecărui itinerar tehnologic.

Comenzile sunt grupate în zile de lucru astfel încât să nu depășească numărul maxim de ore de funcționare a mașinilor, dar această valoare să fie cât mai apropiată de limita superioară. În urma acestui process se pot realiza graficele de producție zilnice sau săptamanale. Această programare a producției permite o utilizare optimizată a resurselor, reducând astfel perioadele de inactivitate ale echipamentelor și minimizând risipa de resurse. A doua procedură se referă la ordonarea proceselor mai complexe, în care fiecare comandă are propriul ei itinerar tehnologic. În aceste cazuri comenzile din set se includ în grafic una câte una, ordinea în care sunt ele considerate fiind stabilită conform unui criteriuadecvat (de exemplu după termenele de livrare sau începând cu comenzile cu cel mai mare număr de operaţii etc).

Când o anumită comandă se programează, graficul conţine deja alte comenzi incluse anterior; ţinând cont de ele se procedează astfel:

1. se determină pentru operaţia i a comenzii în discuţie termenul cel mai devreme, T_i, la care aceasta poate să înceapă. În stabilirea acestui moment trebuie să se ţină seama de două aspecte:

- pe maşina aferentă operaţiei i trebuie să existe, cu începere de la T_i, un interval de timp neocupat suficient de lung pentru a include operaţia i;
- momentul T_i nu trebuie să survină înainte ca operaţiile care preced operaţiai să poată fi executată.

2. se calculează mărimea t_i prin însumarea duratei operaţiei i cu duratele tuturor operaţiilor care-i urmează pe itinerarul tehnologic; t_i reprezintă timpul necesar terminării comenzii dacă aceasta se desfăşoară fără nici o întârziere din momentul în care execuţia a ajuns la operaţia i.

3. se calculează suma (T_i+t_i) pentru toate operaţiile, suma arată momentul în care poate avea loc încheierea prelucrării comenzii, aşa cum acest moment este dictat deoperaţia i.

4. se identifică cea mai mare dintre sumele (T_i+t_i); operaţia aferentă este „locîngust".

5. se include în grafic mai întâi operaţia „loc îngust", cu începere la momentul găsitla punctul 1. operaţiile predecesoare ei se programează cât mai târziu posibil, iar cele succesoare cât mai devreme posibil.

Înainte de a întocmi graficele de execuție, trebuie luate în considerare și alte elemente ale ciclului de fabricație care nu sunt prevăzute în documentația tehnologică standard. De exemplu, timpul necesar pentru deplasarea lotului între două operații în cadrul aceluiași centru de producție sau între centre de producție diferite, timpul alocat pentru controlul tehnic și predarea comenzii la magazie, etc., sunt factori importanți care influențează programarea producției. Acești timpi trebuie calculați și integrați în graficul de execuție pentru a asigura o planificare realistă și fezabilă a producției.

Procesul de construirea unui grafic de execuție este iterativ, ceea ce înseamnă că la includerea unei noi comenzi în grafic, pot apărea necesități de modificare a programării comenzilor anterioare. Acest lucru este deosebit de important în situațiile în care termenele de livrare se modifică, necesitând o rearanjare a priorităților în funcție de noile condiții.

Procedura euristică prezentată nu urmărește explicit un criteriu de optimizare, însă modul în care funcționează contribuie indirect la reducerea stocurilor de producție neterminată și la finalizarea mai rapidă a comenzilor. Prin reducerea timpilor de inactivitate și a întârzierilor, această metodă contribuie la îmbunătățirea eficienței generale a procesului de producție.

3.4.2. Prezentare metode și aplicarea lor (Metode exacte vs metode euristice)

Toate metodele utilizate pentru rezolvarea soluției propuse se bazează fundamental pe modelul matematic dezvoltat de Guinet. Acest model constituie nucleul abordării noastre, oferind un cadru robust și bine fundamentat pentru optimizarea proceselor de producție. Prin aplicarea modelului Guinet, am asigurat o structură metodologică riguroasă reprezentată de funcția obiectiv însoțită de constrângerile și condițiile aferente, care permite identificarea și eliminarea locurilor înguste, precum și ordonarea eficientă a activităților de producție. Astfel, modelul servește ca pilon central în dezvoltarea soluțiilor, garantând o aliniere coerentă și eficientă a strategiilor de optimizare implementate.

Experimentele au fost desfășurate pe un laptop Lenovo, echipat cu un procesor Intel Core Ultra 7, cu 32 GB RAM și SSD de 1 TB, asigurând astfel o performanță optimă pentru procesarea complexă și asigurarea unei viteze de procesare superioare. Pentru dezvoltarea și testarea algoritmilor, s-au utilizat mediile de modelare CPLEX și MATLAB [120], care au permis simularea eficientă și analiza rezultatelor, contribuind la validarea și optimizarea soluțiilor propuse în cadrul studiului.

Analiza detaliată a problemei de programare a producției a relevat complexitatea inerentă a acestui proces și a subliniat necesitatea aplicării unor metode matematice avansate pentru optimizarea timpilor de livrare. În contextul actual, în care cerințele pieței devin din ce în ce mai stringente, iar competiția este acerbă, eficientizarea proceselor de producție reprezintă un element crucial pentru asigurarea competitivității și satisfacerea cerințelor clienților.

Pentru a răspunde acestei provocări, s-a decis utilizarea metodei Guinet, recunoscută pentru eficiența sa în modelarea și optimizarea proceselor de producție complexe. Această metodă a permis

dezvoltarea unei soluții matematice robuste, care a constat în formularea unei funcții obiectiv, alături de identificarea și integrarea unor constrângeri esențiale ale procesului de producție. Funcția obiectiv a avut ca scop principal minimizarea timpului total necesar pentru finalizarea comenzilor, un aspect esențial în contextul unei industrii în care livrarea la timp este un factor determinant pentru satisfacția clienților și pentru menținerea relațiilor comerciale pe termen lung. O prezentare sistematică a prezentei analize ar putea arăta astfel:

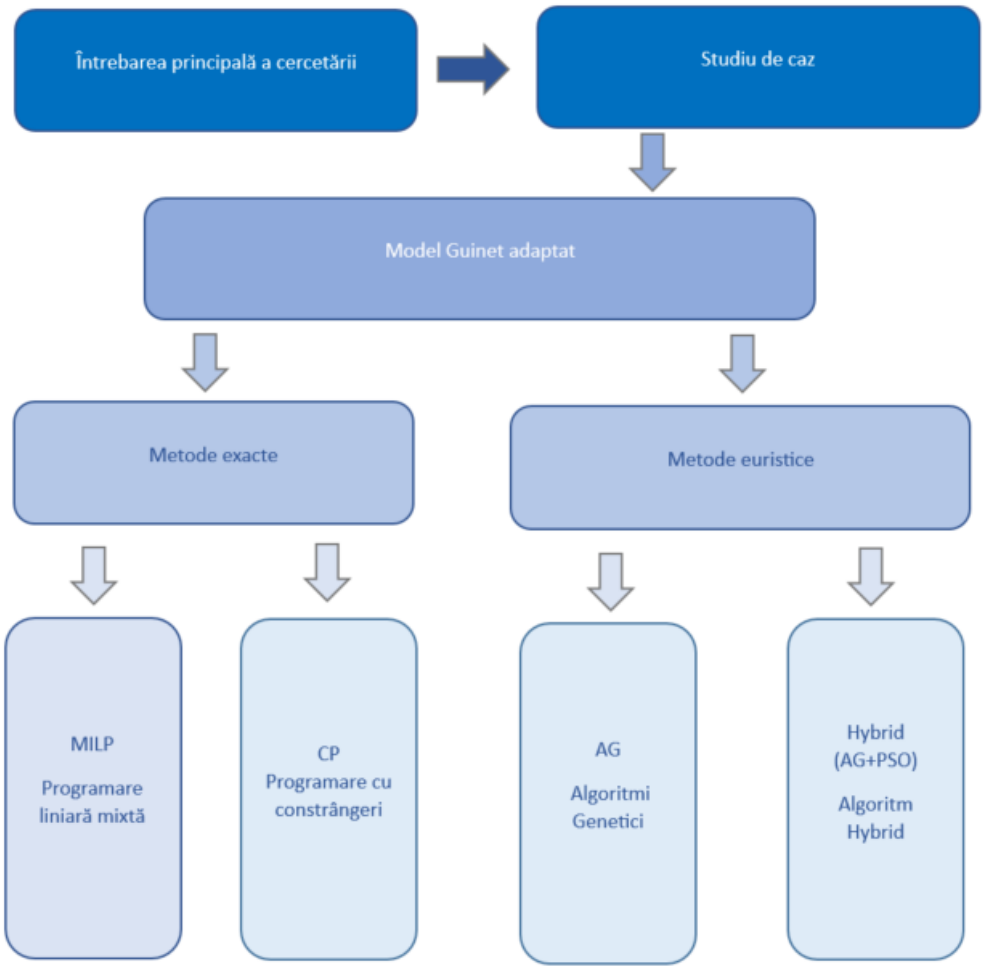


Figura 3.4. Sistematizarea problemei de cercetare – Metode exacte vs Metode euristice (sursa: prelucrare proprie)

Una dintre principalele constrângeri impuse soluției a fost limitarea timpului de producție la maxim șase săptămâni calendaristice. Această cerință a fost esențială pentru respectarea termenelor de livrare și a impus o limită strictă asupra timpului disponibil pentru procesarea comenzilor. În termeni practici, această perioadă de șase săptămâni a fost transformată în valori măsurabile și

concrete, care au inclus un număr de 30 de zile lucrătoare, echivalent cu 480 de ore de lucru, desfășurate în cadrul a două schimburi zilnice de câte 16 ore.

În mod corespunzător, aceste valori au fost detaliate în 28.800 de secunde sau, într-o măsură și mai granulară, 1.728.000 de secunde. Aceste unități de măsură au fost utilizate ca reper de bază pentru evaluarea performanței diverselor metode de rezolvare aplicate, stabilind un standard clar și măsurabil în cadrul procesului de optimizare.

Tabel 3.3. Limitarea timpului de producție

Saptamani	Zile lucratoare	Ore 2*16	Minute	Secunde
6	30	480	28800	1728000

(*sursa: prelucrare proprie)*

Programarea producției este un domeniu complex ce necesită soluții eficiente pentru a optimiza resursele și a respecta termenele limită. În această analiză, vom compara două categorii principale de metode utilizate în optimizarea planificării producției: metodele exacte (utilizând CPLEX/OPL Mixt Integer Linear Programming și CPLEX Constraint Programming) și metodele euristice (algoritmul genetic AG și algoritmul hibrid AG + PSO). [121]

I. Metodele exacte:

1.1) CPLEX/OPL Mixt Integer Linear Programming (MILP)

Metodele exacte, cum ar fi programarea liniară (MILP) utilizată în CPLEX/OPL, se bazează pe modele matematice riguroase pentru a găsi soluții optime. Aceste metode sunt deterministe, ceea ce înseamnă că pentru o anumită problemă și un set de parametri, soluția va fi întotdeauna aceeași. Programarea liniară implică formularea unei funcții obiectiv și a unui set de constrângeri, care sunt rezolvate pentru a găsi valorile optime ale variabilelor de decizie.

1.2) CPLEX Constraint Programming (CP)

Programarea cu constrângeri (CP) este o metodă exactă utilizată pentru a rezolva probleme complexe care nu pot fi formulate ușor în termenii programării liniare. CP utilizează un set de variabile, domenii și constrângeri pentru a modela problema, iar soluția este găsită prin explorarea

sistematică a spațiului soluțiilor posibile. Metodele de CP sunt deosebit de utile pentru problemele combinatorii complexe, cum ar fi planificarea și alocarea resurselor.

II. Metodele Euristice :

2.1) Algoritmul Genetic (AG**)**

Algoritmul Genetic (AG) este o tehnică de optimizare evoluționistă care utilizează procese inspirate din biologia naturală, cum ar fi selecția, crossover-ul și mutația, pentru a evolua soluții la probleme complexe. AG începe cu o populație de soluții posibile (cromozomi), fiecare reprezentând o soluție potențială a problemei. Soluțiile sunt evaluate pe baza unei funcții obiectiv, iar cele mai bune soluții sunt selectate pentru a genera noi soluții prin crossover și mutație.

2.2) Algoritmul Hibrid (AG **+ PSO)**

Algoritmul hibrid combină Algoritmul Genetic (AG) cu Optimizarea prin Roiurile de Particule (PSO) pentru a îmbunătăți performanța și diversitatea soluțiilor. PSO este inspirat de comportamentul colectiv al roiurilor de păsări sau al bancurilor de pești, unde fiecare particulă din populație se mișcă în spațiul soluțiilor, ghidată de propria experiență și de cea a vecinilor săi. Combinarea AG și PSO permite algoritmului hibrid să beneficieze atât de diversitatea genetică oferită de AG, cât și de convergența rapidă oferită de PSO.

3.4.2.1. Metode exacte.

3.4.2.1.1. CPLEX/OPL Mixt Integer Linear Programming (MILP)

Mixt Integer Linear Programming (MILP) în CPLEX OPL [121] implică formularea și rezolvarea problemelor de optimizare unde unele variabile de decizie sunt constrânse să fie numere întregi, în timp ce altele pot lua valori continue. Procesul de rezolvare a unui MILP în CPLEX OPL implică mai multe concepte și etape teoretice esențiale:

1. Modelarea problemei

- **Variabilele de decizie**: În MILP, variabilele de decizie pot fi fie continue (luând orice valoare reală), fie întregi (restricționate la valori întregi). CPLEX OPL permite definirea acestor variabile cu tipuri specifice, cum ar fi int pentru variabilele întregi și float pentru variabilele continue.
- **Funcția Obiectiv**: Scopul MILP este de a optimiza (maximiza sau minimiza) o funcție obiectiv liniară, care este o combinație liniară a variabilelor de decizie. Coeficienții din această funcție reprezintă contribuția fiecărei variabile la obiectiv.

- **Constrângeri**: Problema este supusă unui set de constrângeri liniare. Aceste constrângeri sunt, de asemenea, combinații liniare ale variabilelor de decizie și reprezintă limitările sau cerințele pe care orice soluție fezabilă trebuie să le satisfacă.

2. Formularea MILP

- Formularea MILP în CPLEX OPL constă în definirea funcției obiectiv și a constrângerilor folosind expresii liniare care implică variabilele de decizie. Modelul specifică, de asemenea, care variabile sunt întregi și care sunt continue. Problema este astfel exprimată într-o formă matematică pe care solverul CPLEX o poate înțelege și procesa.

3. Relaxarea și ramificarea

- **Relaxarea**: Una dintre tehnicile fundamentale utilizate în rezolvarea problemelor MILP este relaxarea. Aceasta implică permiterea temporară a variabilelor întregi să ia valori continue (relaxarea constrângerilor de integritate). Problema rezultată, numită relaxare de Programare Liniară (MILP), este mai ușor de rezolvat.
- **Branch-and-Bound**: După rezolvarea problemei relaxate, CPLEX utilizează o metodă numită Branch-and-Bound. [122] Această metodă explorează sistematic spațiul soluțiilor potențiale prin împărțirea problemei în subprobleme (ramificare). În fiecare subproblemă, se adaugă constrângeri suplimentare pentru a se asigura că sunt respectate constrângerile de integritate. Procesul continuă recursiv, explorând regiuni fezabile ale spațiului soluțiilor, până când se găsește o soluție optimă întregă sau toate posibilitățile sunt epuizate.
- **Planuri de tăiere**: CPLEX poate utiliza și planuri de tăiere, care sunt constrângeri suplimentare adăugate pentru a exclude porțiuni din spațiul de căutare care nu conțin soluții întregi. Aceste tăieturi ajută la îmbunătățirea eficienței procesului Branch-and-Bound, reducând numărul de subprobleme care trebuie explorate.

4. Căutarea soluției optime

- Solverul MILP din CPLEX folosește o combinație a tehnicilor menționate pentru a căuta o soluție optimă. Solverul rafinează iterativ regiunea fezabilă prin ramificare și adăugare de tăieturi, verificând în mod constant relaxările LP în fiecare nod. Valoarea obiectivului este evaluată în fiecare nod, iar cea mai bună soluție întreagă găsită este actualizată.

- **Pruning (Tăierea Ramurilor)**: În Branch-and-Bound, anumite ramuri (subprobleme) pot fi ignorate dacă nu pot oferi o soluție mai bună decât cea curentă sau dacă nu satisfac constrângerile.

5. Terminarea și soluția
   - Algoritmul se termină atunci când toate ramurile au fost explorate sau tăiate, iar cea mai bună soluție fezabilă întregă este identificată ca soluția optimă. CPLEX furnizează apoi această soluție, incluzând valorile variabilelor de decizie și valoarea obiectivului corespunzător.
   - **Dualitate și sensibilitate**: Analiza post-soluție în MILP implică adesea înțelegerea variabilelor duale asociate cu constrângerile și sensibilitatea soluției la modificările parametrilor problemei. Totuși, variabilele duale sunt mai puțin clare în MILP comparativ cu problemele LP pure datorită naturii discrete a variabilelor întregi.
6. Scalabilitate și complexitate
   - Problemele MILP sunt, în general, dificile, ceea ce înseamnă că complexitatea computațională poate crește exponențial odată cu mărimea problemei. Cu toate acestea, CPLEX este conceput pentru a gestiona eficient problemele MILP de mare anvergură folosind algoritmi și euristici sofisticate. Performanța depinde de structura problemei, de calitatea formulării modelului și de capacitatea solverului de a exploata caracteristicile specifice ale problemei.

În rezumat, rezolvarea problemelor MILP în CPLEX OPL implică formularea problemei cu o combinație de variabile întregi și continue, aplicarea unor tehnici precum relaxarea LP, Branch-and-Bound și planurile de tăiere pentru a naviga în spațiul soluțiilor, și rafinarea iterativă a căutării până la găsirea unei soluții optime.

Pentru mai multe detalii despre componentele cheie ale CPLEX/OPL LP precum și explicațiile privind limitările oferite de acesta, a se vedea Anexa 2.

Cel mai important neajuns al acestei abordări este timpul de rulare până la găsirea unei soluții. Acest lucru se datorează modului ce căutare a soluției optime din varietatea posibilelor soluții.

În varianta dezvoltată cu tehnica MILP, software-ul a rulat timp de peste 29 ore fără a genera o soluție din cauza complexității matricei implicate.- trimitere la Anexele noi. Matricea era compusă din numărul de comenzi (280), numărul de produse (353) și numărul de mașini pe care acestea trebuiau procesate (51), rezultând într-un număr imens de variante de soluții (5040840). Acest lucru evidențiază limitările metodelor exacte pentru probleme de mari dimensiuni și complexitate.

În cazul metodei MILP implementată prin CPLEX OPL, s-a observat că complexitatea matematică și dimensiunea problemei au depășit capacitatea acestui model de a oferi o soluție

practică. Incapacitatea programul de a livra o soluție după aproximativ 29 ore de funcționare(menționată în figura următoare) continuă, a condus la concluzia că metoda este inadecvată pentru probleme de această natură, cel puțin în forma sa actuală.

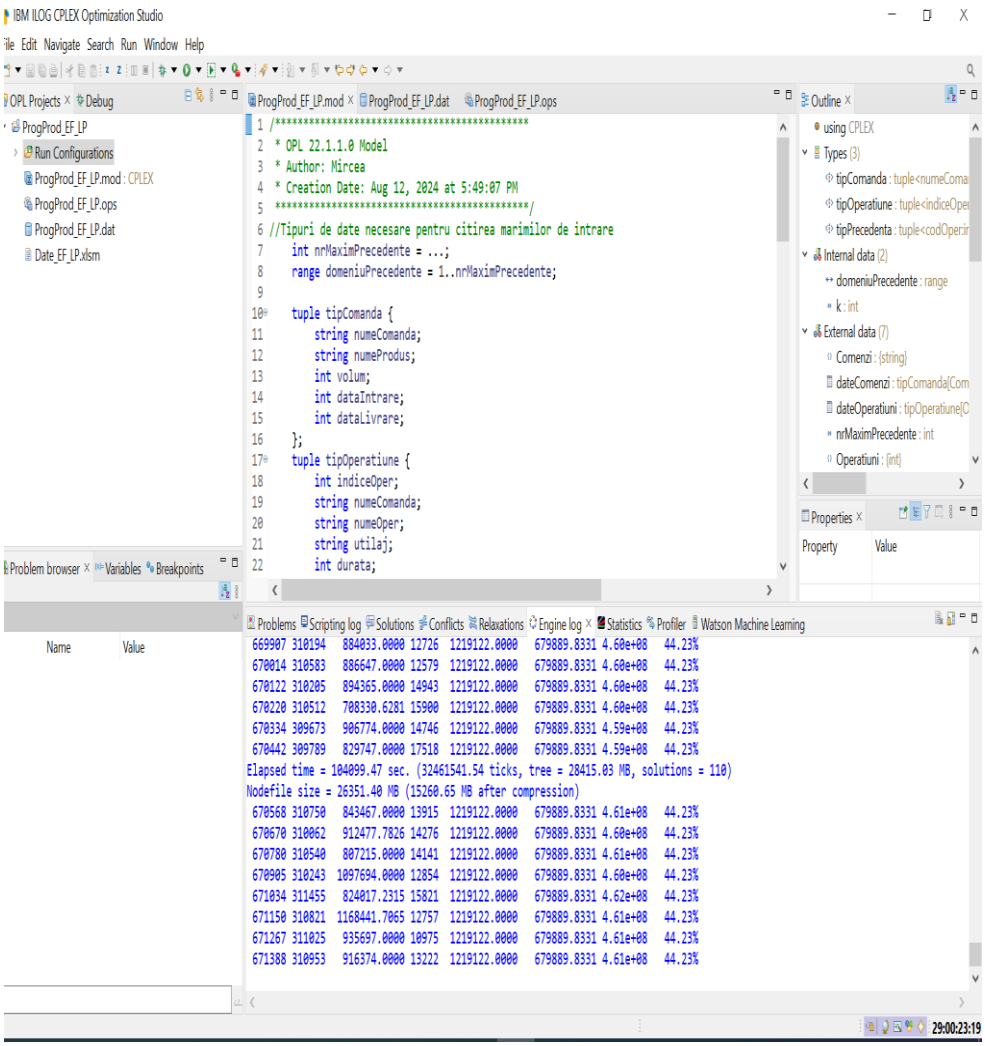


Figura 3.5. Captură privind anexa statisticilor de rulare MILP
(sursa: prelucrare proprie)

În ciuda faptului ca procesorul laptopului era folosit în proporție 81,6 % de CPLEX (figura următoare), acesta nu a putut oferi o soluție generînd un mesaj de eroare (figura privind captura mesajului de eroare).

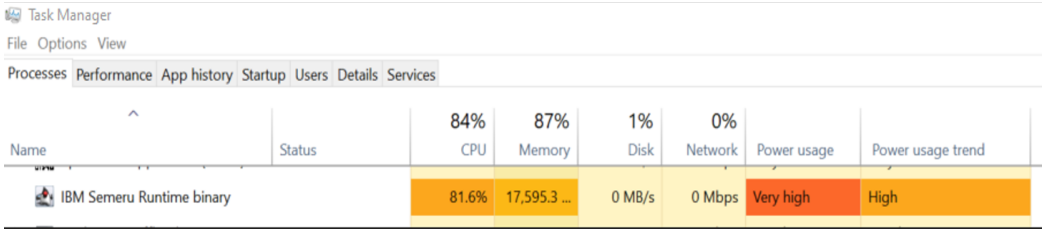


Figura 3.6. Captură privind uzura procesor CPLEX MILP

(sursa: prelucrare proprie)

Aceasta (figura) sugerează că, în probleme complexe care implică un număr mare de variabile și constrângeri, MILP poate deveni ineficientă sau chiar impracticabilă. În tabelul de mai jos sunt sintetizate datele principale ale acestei variante.

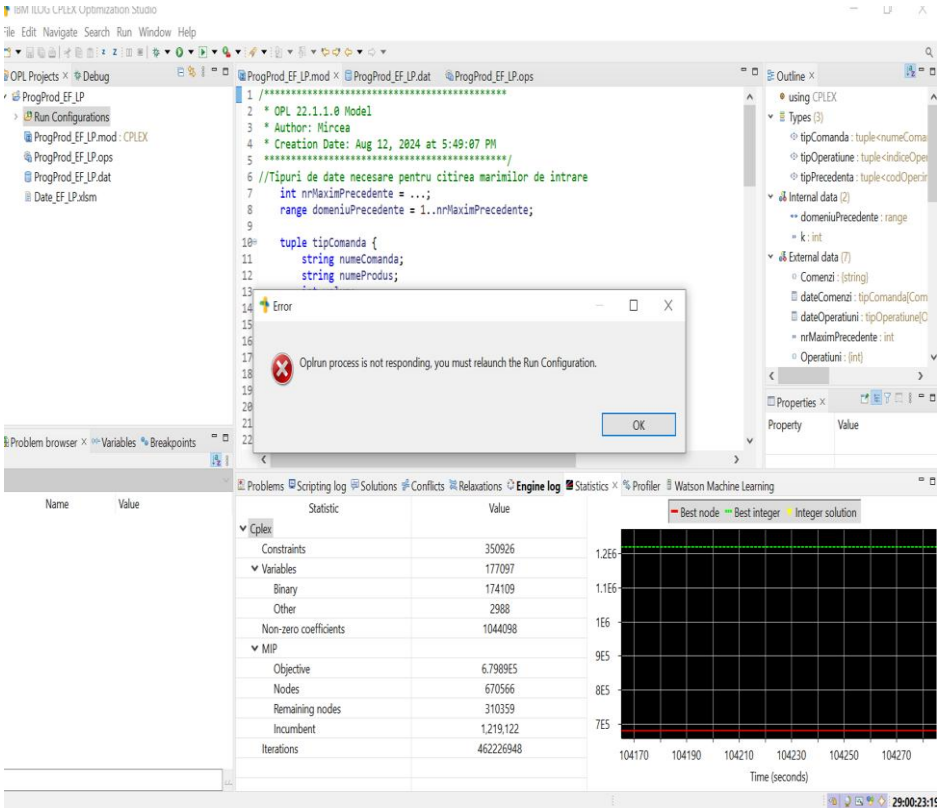


Figura 3.7. Captură privind mesaj eroare CPLEX MILP
(sursa: prelucrare proprie)

Tabel 3.4. Rezultate Cplex Mixt Integer Linear Programming

Cplex Mixt Integer Linear Programming	Ciclu de productie contractual					Ciclu de productie obtinut		
	Sapt.	Zile	Ore 2*16	Min.	Sec.	Ore	Sec.	Timp rulare
	6	30	480	28800	1728000	ned.	ned.	>104400

(sursa: prelucrare proprie)

Din acest motiv, s-a explorat utilizarea Software-ul CPLEX/OPL Constrain Programming (CP), care oferă o abordare mai eficientă pentru rezolvarea problemelor complexe. CPLEX/OPL CP poate gestiona complexitatea matricei în mod diferit și poate oferi soluții optime într-un timp rezonabil, îmbunătățind semnificativ eficiența procesului decizional. Prin aplicarea CPLEX/OPL CP, s-a reușit reducerea timpului de calcul și găsirea unei soluții viabile, demonstrând avantajele acestei metode alternative în comparație cu metodele tradiționale de programare liniară.

3.4.2.1.2. CPLEX/OPL Constrain Programing (CP)

În contextul operațional al unei companii, deciziile privind orizontul de programare al activităților de producție sunt esențiale pentru eficiența și adaptabilitatea organizației. Planificarea corectă a resurselor și a producției poate influența direct costurile, calitatea produselor și capacitatea de a răspunde rapid la cerințele pieței. În acest context, analizarea numărului comenzilor și ajustarea orizontului de programare devin instrumente esențiale pentru îmbunătățirea performanței operaționale.

Orizontul de programare reprezintă intervalul de timp pe care o companie îl consideră atunci când își planifică activitățile de producție. Un orizont mai lung oferă o mai mare predictibilitate și permite companiei să se pregătească pentru cererile viitoare. Totuși, acesta poate duce la o lipsă de flexibilitate și la dificultăți în adaptarea la schimbările rapide din piață. În schimb, un orizont mai scurt permite o mai mare agilitate, dar necesită o monitorizare constantă și capacitatea de a reacționa rapid la modificările cerințelor clienților.

Această abordare era benefică într-o piață stabilă, cu o cerere previzibilă, permițând companiei să își optimizeze fluxurile de aprovizionare și să minimizeze costurile prin comenzi la termen și producție în loturi mari. Totuși, analiza numărului de comenzi a indicat o tendință spre o variabilitate mai mare în cerere, sugerând o dinamică a pieței ce necesită o flexibilitate și capacitate de adaptare sporite. Pe măsură ce cererea devine din ce în ce mai dificil de anticipat pe termen lung, un orizont de programare de mare poate deveni un dezavantaj, împiedicând compania să răspundă prompt la cererile fluctuante și să optimizeze stocurile.

Decizia de reducere a orizontului de programare. Ca răspuns la această schimbare în dinamică, s-a luat decizia de a reduce orizontul de programare la 3 săptămâni. Această schimbare strategică reflectă o ajustare semnificativă în modul de operare al companiei, cu implicații asupra gestionării resurselor, producției și relațiilor cu furnizorii și clienții.

Reducerea orizontului de programare la 3 săptămâni înseamnă că toate activitățile de producție și aprovizionare vor fi planificate pe o perioadă mai scurtă, permițând companiei să se adapteze mai rapid la schimbările din cerere. Această abordare oferă mai multe avantaje:

- cu un orizont de programare mai scurt, compania poate ajusta rapid planurile de producție și aprovizionare în funcție de cerințele pieței. Aceasta permite o mai bună gestionare a stocurilor și o reducere a riscului de suprastocare sau de lipsuri în stocuri.

- într-o piață dinamică, capacitatea de a răspunde rapid la cerințele fluctuante ale clienților este esențială. un orizont de programare de 3 săptămâni permite companiei să fie mai reactivă și să îmbunătățească serviciul oferit clienților, menținând în același timp un nivel ridicat de satisfacție.
- într-un mediu de afaceri volatil, prognozele pe termen lung pot fi mai puțin fiabile. Prin reducerea orizontului de programare, compania își reduce expunerea la riscurile asociate cu schimbările bruște din piață, cum ar fi fluctuațiile în cerere sau modificările în preferințele consumatorilor.
- un orizont de programare mai scurt permite o programare mai precisă a utilizării resurselor, reducând risipa și îmbunătățind eficiența operațională. Acest lucru poate contribui la reducerea costurilor operaționale prin optimizarea fluxurilor de aprovizionare și producție.

Implementarea schimbării. Implementarea deciziei de a reduce orizontul de programare necesită o revizuire a mai multor procese interne. În primul rând, compania trebuie să își ajusteze sistemele de programare și control al producției pentru a funcționa eficient pe intervale mai scurte. Aceasta poate implica adoptarea unor noi tehnologii și sisteme de management al informației care să permită o monitorizare mai precisă și în timp real a cererii și a resurselor disponibile. De asemenea, va fi necesară o colaborare strânsă cu furnizorii pentru a asigura că aceștia pot răspunde prompt la comenzile mai frecvente, dar de dimensiuni mai mici. În unele cazuri, aceasta poate necesita renegocierea termenilor contractuali sau identificarea de noi furnizori care pot oferi mai multă flexibilitate.

În ceea ce privește producția, compania va trebui să își ajusteze capacitatea de producție pentru a se alinia cu noul orizont de programare. Aceasta poate implica modificări ale planurilor de producție, inclusiv creșterea frecvenței schimbărilor de producție sau reducerea dimensiunii loturilor produse. Este posibil să fie necesară o formare suplimentară pentru angajați, pentru a se asigura că aceștia sunt pregătiți să lucreze într-un mediu de producție mai dinamic și mai flexibil.

În timp ce reducerea orizontului de programare oferă numeroase beneficii, aceasta vine și cu anumite provocări. Gestionarea unui orizont de programare mai scurt necesită o mai mare disciplină operațională și capacitatea de a lua decizii rapide pe baza datelor disponibile. Compania va trebui să investească în tehnologii de colectare și analiză a datelor, pentru a asigura că are acces la informațiile necesare pentru a lua decizii bine informate.

De asemenea, un orizont de programare mai scurt poate crește presiunea asupra echipei de management și asupra personalului de producție, care va trebui să se adapteze la un ritm mai rapid de schimbări și la o cerere mai mare de flexibilitate. Aceasta poate necesita o reevaluare a structurilor de recompensare și motivare pentru a menține moralul și angajamentul angajaților.

Un alt aspect de luat în considerare este impactul asupra relațiilor cu furnizorii. Reducerea orizontului de programare poate duce la comenzi mai frecvente, dar de dimensiuni mai mici, iar compania trebuie să se asigure că furnizorii sunt capabili să răspundă la aceste cerințe fără a compromite calitatea sau termenii de livrare. În unele cazuri, poate fi necesară dezvoltarea unor parteneriate strategice cu furnizorii pentru a asigura alinierea obiectivelor și a capacităților de producție.

Reducerea orizontului de programare la 3 săptămâni reprezintă o decizie strategică menită să îmbunătățească flexibilitatea și capacitatea de reacție a companiei într-un mediu de afaceri dinamic și imprevizibil. Această schimbare reflectă nevoia de a răspunde rapid la cerințele pieței și de a optimiza resursele pentru a menține competitivitatea și eficiența operațională.

În implementarea acestei schimbări, compania trebuie să ia în considerare atât beneficiile, cât și provocările asociate, asigurându-se că are resursele și procesele necesare pentru a funcționa eficient în noul cadru operațional. Investițiile în tehnologie, formarea personalului și colaborarea cu furnizorii vor fi esențiale pentru succesul acestei tranziții și pentru atingerea obiectivelor pe termen lung ale organizației.

Analiza modelului de optimizare și a soluțiilor utilizând IBM ILOG CPLEX Optimization Studio - Înțelegerea modelului de optimizare. Modelul (a se vedea **ANEXA 3**) definește o problemă de programare utilizând funcțiile de Programare cu Constrângeri (CP) ale CPLEX. O descriere detaliată a componentelor și rolurilor acestora ar putea fi rezumată astfel:

a) Structuri de date (Tuples) și intervale

- tipComanda: Reprezintă o comandă cu atribute precum numele comenzii, numele produsului, volumul, data de început și data de livrare.
- tipOperatiune: Reprezintă o operațiune cu atribute precum indexul operațiunii, numele comenzii, numele operațiunii, utilajul și durata.
- tipPrecedenta: Reprezintă relațiile de precedență între operațiuni.

b) Seturi și inițializarea datelor

- Seturi: Definește colecții pentru comenzi (Comenzi), operațiuni (Operatiuni), utilaje (setUtilaje) și precedențe (setPrecedente).
- Inițializarea datelor: Datele sunt citite dintr-un fișier Excel (Date_EF.xlsm), incluzând detalii despre comenzi, operațiuni și constrângeri de precedență.

c) Variabile de decizie

- intOperatiuni: Variabile de decizie de interval care reprezintă momentele de început și sfârșit ale fiecărei operațiuni.

- conflicte: Variabile de decizie de secvență pentru gestionarea conflictelor de operațiuni pe utilaje.
- inceputUtilizare și sfarsitUtilizare: Variabile de decizie întregi pentru momentele de început și sfârșit ale utilizării utilajelor.
- inceputComanda și sfarsitComanda: Variabile de decizie întregi pentru momentele de început și sfârșit ale comenzilor.

d) Funcția obiectiv

Scopul este de a minimiza timpul total de producție, adică timpul de finalizare al ultimei operațiuni. Aceasta se realizează prin minimizarea timpului de finalizare maxim al tuturor operațiunilor.

Constrângeri ale funcției obiectiv:

- **Constrângeri de început și sfârșit**: Asigură că operațiunile încep după data de început a comenzii și se termină înainte de data de livrare a comenzii.
- **Constrângeri de precedență**: Asigură că operațiunile urmează ordinea specificată.
- **Constrângeri de non-suprapunere**: Asigură că două operațiuni pe același utilaj nu se suprapun.
- **Constrângeri de utilizare**: Definirea momentelor de început și sfârșit pentru utilizarea utilajelor.
- **Constrângeri pentru comenzi**: Definirea momentelor de început și sfârșit pentru comenzi pe baza operațiunilor.

Explicarea detaliată a softului - cum funcționează CP în OPL Aplicat

În cercetare, utilizăm OPL (Optimization Programming Language), un limbaj de programare pentru optimizare. Primul fișier cu extensia .mod este modelul matematic formulat în limbajul OPL. Acesta conține variabile și structuri care facilitează preluarea datelor și calcularea soluției prin intermediul OPL. La începutul fișierului, apare comanda “using CP”, o instrucțiune ce specifică utilizarea Programării cu Constrângeri (CP). Aceasta înseamnă că programul și funcția obiectiv sunt rezolvate printr-o serie de funcții specifice OPL pentru optimizarea programării producției.

Motorul CP, dezvoltat de IBM, este conceput pentru a facilita scrierea și rezolvarea problemelor de programare a producției. Structura fișierului .mod include zone cu structuri de date specifice unui model bazat pe CP, cum ar fi variabilele de tip interval și secvență, care simplifică scrierea constrângerilor.

Constrângerile sunt formulate aproape natural în acest program. De exemplu, funcția endBeforeStart din capitolul constrângerilor specifică că o activitate trebuie să se încheie înainte de a începe alta. Acest mod de rezolvare funcționează mult mai rapid decât programarea liniară tradițională pentru astfel de activități, datorită acronimelor specifice dezvoltate pentru a rezolva modelul mai eficient. În concluzie, Programarea cu Constrângeri (CP) în OPL oferă o abordare mai eficientă pentru rezolvarea problemelor complexe de programare a producției, reducând semnificativ timpul necesar pentru a ajunge la soluții optime. Această metodă este deosebit de utilă în gestionarea complexității și obținerea rapidă a rezultatelor dorite.

În această secțiune urmează o analiză a tipurilor de date utilizate în modelul de optimizare. În primul rând, se definește o valoare întreagă a numărului maxim de precedențe, a cărei declarație, după semnul egal, conține trei puncte. Aceasta indică faptul că valorile sunt preluate dintr-un alt fișier, anume al doilea fișier .dat (a se vedea Anexa 3), care conține informația relevantă. Aceasta este a treia informație din fișierul Date_EF.xlsm (a se vedea tabelul următor) unde se află o instrucțiune ce citește numărul maxim de precedențe din fișierul de date.

Tabel 3.5. Datele de intrare reprezentând comenzile furnizorului si termenul de livrare maxim pentru fiecare comandă.

Nr crt	Part Code ABCs	Qty	duedate
C1	ABC 002	60	22/05/23
C95	ABC 051	540	26/05/23
C79	ABC 052	308	25/05/23
C12	ABC 038	50	22/05/23
C70	ABC 311	303	25/05/23
C61	ABC 313	396	25/05/23
C4	ABC 004	76	22/05/23
C5	ABC 006	30	22/05/23
C6	ABC 004	76	22/05/23
C7	ABC 010	30	22/05/23
C267	ABC 313	396	01/06/23
C9	ABC 011	40	22/05/23
C279	ABC 065	150	05/06/23
C14	ABC 283	260	23/05/23

C15	ABC 283	208	23/05/23
C110	ABC 242	420	27/05/23
C135	ABC 067	60	29/05/23
C18	ABC 283	72	24/05/23
C136	ABC 068	40	29/05/23
C20	ABC 283	48	24/05/23
C21	ABC 012	57	24/05/23
C22	ABC 012	57	24/05/23
C23	ABC 016	27	24/05/23
C24	ABC 016	24	24/05/23
C25	ABC 029	60	24/05/23
C26	ABC 015	457	24/05/23
C58	ABC 243	840	25/05/23
C66	ABC 248	180	25/05/23
C29	ABC 009	28	25/05/23
C54	ABC 109	385	25/05/23
C93	ABC 053	500	26/05/23
C32	ABC 009	28	25/05/23
C33	ABC 014	28	25/05/23
C45	ABC 107	2,600	25/05/23
C71	ABC 112	25	25/05/23
C111	ABC 109	330	27/05/23
C201	ABC 142	240	01/06/23

(sursa: prelucrare proprie)

O altă informație importantă la începutul tipului de date este o constantă care asociază un domeniu de valori de la 1 până la valoarea maximă a numărului de precedențe.

Urmează trei structuri de date, denumite "tuple", care sunt uniuni de date. Prima structură se numește tipComanda, a doua tipOperatiune și a treia tipPrecedenta. În prima structură, tipComanda, se regăsesc cinci atribute: primele două sunt șiruri de caractere, iar următoarele sunt valori întregi. Din fișierul Excel, aceste valori sunt citite dintr-un tabel care conține coloanele: nume comandă, nume produs, volum, data de intrare și data de livrare. În mod similar, aceeași logică se aplică și celorlalte două structuri de date.

Odată ce aceste structuri sunt definite, ele pot fi utilizate în următoarele trei instrucțiuni, care sunt declarații. De exemplu, *Comenzi* este asociat unui tablou ale cărui elemente sunt șiruri de caractere. După semnul egal apar cele trei puncte, indicând că valorile sunt stabilite în fișierul .dat. În fișierul de date, există comanda sheetConnection, care face legătura între setul de date și fișierul Excel. Comanda sheetRead este utilizată pentru a citi date dintr-o anumită zonă a fișierului Excel, iar

comanda sheetWrite este folosită pentru a exporta date într-o anumită zonă din sheet-ul numărul 4, unde se prezintă soluția.

Prin aceste două comenzi, se stabilește o metodă eficientă și elegantă pentru preluarea și exportul valorilor. Constanta DateComenzi este un tablou care conține elemente de tipul structurilor definite anterior. Astfel, un element din acest tablou conține cinci valori de tip *tipComanda*, legate de tabelele din fișierele Excel. Această metodă permite citirea eficientă a unui set mare de valori. Accesarea unui element din tablou se face folosind numele comenzii ca indice, ceea ce face referința și diferențierea elementelor din tabel mult mai intuitive.

În concluzie, această structură complexă de date și metodele de preluare și export ale acestora permit o gestionare eficientă și precisă a informațiilor necesare pentru optimizarea procesului de producție.

Fișierul Excel date_ef.xlsm este un fișier cu macro-uri, ceea ce înseamnă că include cod Visual Basic (VB) dezvoltat pentru sheet-ul 1. Acest cod ne permite să organizăm și să utilizăm datele din acest fișier în scopul preluării lor de către modelul dezvoltat în CPLEX. Detalii despre codul VB sunt prezentate în tabelul următor.

Tabel 3.6. Modul în care datele de intrare au fost aranjate după implementarea scriptului Visual Basic.

Lista comenzilor de programat					Procese tehnologice - toate produsele				Lista operatiunilor de programat				
Comenzi	Produs	Cantitate	Dispon.	Livrare	Produs	Nume operatiune	Utilaj	Durata [sec/buc]	Ind oper.	Comanda	Nume operatiune	Utilaj	Durata [sec/com]
C1	AMT000218-02-SEF	60	0	1728000	2070786-2EF	Oper1	M1	3	1	C1	Oper396	M10	4980
C2	SEM102154-01-SEF	304	0	1728000	2070786-2EF	Oper2	M15	6	2	C1	Oper397	M15	720
C3	TF50074667	44	0	1728000	2070786-2EF	Oper3	M23	19	3	C1	Oper398	M16	1800
C4	TF50077149	76	0	1728000	2070786-2EF	Oper4	M44	6	4	C1	Oper399	M18	900
C5	TF50133716	30	0	1728000	2070786EF	Oper5	M1	146	5	C1	Oper400	M20	1080
C6	TF50077149	76	0	1728000	2070786EF	Oper6	M15	6	6	C1	Oper401	M34	2880
C7	TF50134486	30	0	1728000	2070786EF	Oper7	M22	144	7	C1	Oper402	M43	360
C8	TF50161270	72	0	1728000	2070786EF	Oper8	M23	90	8	C1	Oper403	M44	360
C9	AGSC73138-01-SEF	40	0	1728000	2070786EF	Oper9	M44	6	9	C2	Oper1780	M10	6384
C10	TF50081822	88	0	1728000	2070786-SEF	Oper10	M1	146	10	C2	Oper1781	M11	33744
C11	TF50134457	30	0	1728000	2070786-SEF	Oper11	M15	6	11	C2	Oper1782	M15	3648
C12	EIBAD1311-01-SEF	50	0	1728000	2070786-SEF	Oper12	M22	144	12	C2	Oper1783	M16	20064
C13	BA028054-SEF	1600	0	1728000	2070786-SEF	Oper13	M23	90	13	C2	Oper1784	M18	4560
C14	LA033185-SEF	260	0	1728000	2070786-SEF	Oper14	M44	6	14	C2	Oper1785	M20	5472
C15	LA033185-SEF	208	0	1728000	2070787-2EF	Oper15	M15	6	15	C2	Oper1786	M44	1824
C16	GR0019001EF	140	0	1728000	2070787-2EF	Oper16	M23	19	16	C3	Oper1814	M10	1320
C17	BA028053-SEF	6	0	1728000	2070787-2EF	Oper17	M45	6	17	C3	Oper1815	M15	528
C18	LA033185-SEF	72	0	1728000	2070787-2SEF	Oper18	M15	6	18	C3	Oper1816	M16	792
C19	BA028053-SEF	6	0	1728000	2070787-2SEF	Oper19	M23	19	19	C3	Oper1817	M18	528
C20	LA033185-SEF	48	0	1728000	2070787-2SEF	Oper20	M45	6	20	C3	Oper1818	M34	1056
C21	2070790A01-SEF	57	0	1728000	2070787EF	Oper21	M1	201	21	C3	Oper1819	M44	264
C22	2070790A01-SEF	57	0	1728000	2070787EF	Oper22	M15	6	22	C4	Oper1820	M50	456
C23	2070790A02-SEF	27	0	1728000	2070787EF	Oper23	M22	204	23	C5	Oper1822	M50	180
C24	2070790A02-SEF	24	0	1728000	2070787EF	Oper24	M23	90	24	C6	Oper1820	M50	456
C25	2071925A02-SEF	60	0	1728000	2070787EF	Oper25	M44	6	25	C7	Oper1824	M50	180
C26	2072367-SEF	457	0	1728000	2070787-SEF	Oper26	M1	213	26	C8	Oper1825	M8	1296
C27	GR0020001EF	72	0	1728000	2070787-SEF	Oper27	M15	6	27	C8	Oper1826	M15	864
C28	GR0020001EF	96	0	1728000	2070787-SEF	Oper28	M22	144	28	C8	Oper1827	M18	864
C29	2070790A01-KEF	28	0	1728000	2070787-SEF	Oper29	M23	87	29	C8	Oper1828	M19	432
C30	GN0058001-KEF	480	0	1728000	2070787-SEF	Oper30	M44	6	30	C8	Oper1829	M20	1080
C31	A85659-1-F	16	0	1728000	2070790A01EF	Oper31	M50	6	31	C8	Oper1830	M26	2736
C32	2070790A01-KEF	28	0	1728000	2070790A01-KEF	Oper32	M9	90	32	C8	Oper1831	M34	6480
C33	2070790A02-KEF	28	0	1728000	2070790A01-KEF	Oper33	M15	18	33	C8	Oper1832	M44	432
C34	GN0008001-KEF	240	0	1728000	2070790A01-KEF	Oper34	M16	210	34	C9	Oper347	M10	920
C35	GN0016001-KEF	468	0	1728000	2070790A01-KEF	Oper35	M18	30	35	C9	Oper348	M15	480
C36	GN0017001-KEF	500	0	1728000	2070790A01-KEF	Oper36	M22	282	36	C9	Oper349	M18	480
C37	GN0065001-KEF	180	0	1728000	2070790A01-KEF	Oper37	M27	213	37	C9	Oper350	M20	720
C38	GN0005001-KEF	240	0	1728000	2070790A01-KEF	Oper38	M45	6	38	C9	Oper351	M44	240
C39	GR2009001EF	165	0	1728000	2070790A01-KPEF	Oper39	M9	90	39	C10	Oper1821	M50	528
C40	GR2012001EF	690	0	1728000	2070790A01-KPEF	Oper40	M15	18	40	C11	Oper1823	M50	180
C41	GR0026001EF	165	0	1728000	2070790A01-KPEF	Oper41	M16	210	41	C12	Oper526	M10	1550
C42	GR2008001EF	152	0	1728000	2070790A01-KPEF	Oper42	M18	114	42	C12	Oper527	M15	600
C43	GR2011001EF	527	0	1728000	2070790A01-KPEF	Oper43	M22	282	43	C12	Oper528	M18	750
C44	GR0026001EF	165	0	1728000	2070790A01-KPEF	Oper44	M27	213	44	C12	Oper529	M20	900

Explicația detaliată a soluției - Capturile de ecran demonstrează modul în care CPLEX gestionează problema de optimizare, oferind statistici detaliate și valoarea obiectivului rezultat.

Comanda	Operatiune	Inceput	Sfarsit	Utilaj	Utilaj	Inceput utilizare	Sfarsit utilizare	Comanda	Inceput	Sfarsit	Suma dur. oper. [sec]
C1	Oper396	420975	425955	M10	M1	0	583191	C1	420975	1087157	13080
C1	Oper397	722771	723491	M15	M2	0	286202	C2	9146	163305	75696
C1	Oper398	1078187	1079987	M16	M3	-1	1	C3	77001	1087421	4488
C1	Oper399	1080392	1081292	M18	M4	-1	1	C4	21288	21744	456
C1	Oper400	1081715	1082795	M20	M5	0	552484	C5	7896	8076	180
C1	Oper401	1082795	1085675	M34	M6	0	526786	C6	19344	19800	456
C1	Oper402	1085675	1086035	M43	M7	0	589611	C7	2568	2748	180
C1	Oper403	1086797	1087157	M44	M8	0	579115	C8	30606	99465	14184
C2	Oper1780	9146	15530	M10	M9	0	518098	C9	846	18819	2840
C2	Oper1781	48426	82170	M11	M10	0	429442	C10	19800	20328	528
C2	Oper1782	82170	85818	M15	M11	5166	422775	C11	168	348	180
C2	Oper1783	116949	137013	M16	M12	-1	1	C12	72117	381668	8600
C2	Oper1784	151449	156009	M18	M13	285222	288342	C13	51192	271414	102400
C2	Oper1785	156009	161481	M20	M14	25757	532324	C14	404056	767331	89440
C2	Oper1786	161481	163305	M44	M15	36	768424	C15	350162	845516	71552
C3	Oper1814	77001	78321	M10	M16	54	1091321	C16	415898	803196	37660
C3	Oper1815	358128	358656	M15	M17	90	999668	C17	49536	278170	2322
C3	Oper1816	1082417	1083209	M16	M18	0	1091705	C18	574939	1061916	24768
C3	Oper1817	1083209	1083737	M18	M19	318	1090901	C19	49824	272655	2322
C3	Oper1818	1085675	1086731	M34	M20	11016	1083515	C20	219990	714539	16512
C3	Oper1819	1087157	1087421	M44	M21	-1	1	C21	15570	145908	47367
C4	Oper1820	21288	21744	M50	M22	11529	631041	C22	2520	60147	47367
C5	Oper1822	7896	8076	M50	M23	11289	1087045	C23	28350	190851	22113
C6	Oper1820	19344	19800	M50	M24	11799	1085537	C24	23670	100713	19656
C7	Oper1824	2568	2748	M50	M25	292971	293181	C25	7650	76950	49140
C8	Oper1825	30606	31902	M8	M26	12279	973396	C26	61436	454410	32904
C8	Oper1826	65430	66294	M15	M27	20265	1023612	C27	506938	1024044	21672
C8	Oper1827	66294	67158	M18	M28	-1	1	C28	445286	808392	28896
C8	Oper1828	67158	67590	M19	M29	598389	1059324	C29	25830	355514	23772
C8	Oper1829	67590	68670	M20	M30	319581	1078284	C30	371340	958116	54720
C8	Oper1830	75772	78508	M26	M31	11499	637989	C31	52122	324845	11840
C8	Oper1831	92553	99033	M34	M32	366	1089409	C32	0	26397	23772
C8	Oper1832	99033	99465	M44	M33	12464	1061484	C33	13050	82638	23436
C9	Oper347	846	1766	M10	M34	11115	1091501	C34	424718	723377	54720
C9	Oper348	5394	5874	M15	M35	-1	1	C35	129824	793049	130572
C9	Oper349	17379	17859	M18	M36	-1	1	C36	383988	991356	78500
C9	Oper350	17859	18579	M20	M37	-1	1	C37	200444	1030092	49500
C9	Oper351	18579	18819	M44	M38	44859	731147	C38	336402	666521	45840
C10	Oper1821	19800	20328	M50	M39	-1	1	C39	17550	79491	12375
C11	Oper1823	168	348	M50	M40	-1	1	C40	31512	109493	35190
C12	Oper526	72117	73667	M10	M41	-1	1	C41	14754	361522	11715
C12	Oper527	369710	370310	M15	M42	-1	1	C42	721859	1084667	9576
C12	Oper528	375218	375968	M18	M43	100227	1086035	C43	144800	961278	26350
C12	Oper529	375968	376868	M20	M44	982	1091801	C44	3264	149322	11715

Figura 3.8. Captură de ecran privind modul în care datele de ieșire au fost exportate in Excell după rularea modelului realizat in CPLEX OPL Constrain Programing
(sursa: prelucrare proprie)

Informațiile cheie din capturile de ecran pot fi rezumate astfel**:**

- **Valoarea obiectivului**: Valoarea finală a obiectivului este 1.091.828 secunde reprezentând timpul minimizat de finalizare.
- **Numărul de variabile și constrângeri**:
    - **Variabile**: 2.224 variabile de decizie.
    - **Constrângeri**: 4.997 constrângeri, indicând o problemă complexă de optimizare.
- **Utilizarea memoriei**: Utilizarea memoriei este de 171,866 MB, sugerând că dimensiunea problemei este gestionabilă pe hardware modern.
- **Performanța soluției**:
    - **Numărul de ramuri**: 9.835.655 ramuri, indicând amploarea spațiului de căutare explorat de capacitatea de soluționare.
    - **Timpul petrecut**: Găsirea soluției a durat 12,55 secunde în total, din care 2,65 secunde pentru extragerea datelor.

Următoarul tabel de statistici conține:

- **Iterații și ramuri** – care prezintă progresul capacității de a găsi soluția, inclusiv numărul de eșecuri, ramuri și cele mai bune valori obiectiv găsite în fiecare etapă.
- **Utilizarea memoriei** - Reiterează utilizarea memoriei și oferă informații detaliate despre performanța capacității de a găsi soluția.

Comanda	Operatiune	Inceput	Sfarsit	Utilaj	Cod utilaj
C204	Oper724	0	77280	M1	1
C242	Oper934	77280	81264	M1	1
C275	Oper152	81264	103824	M1	1
C224	Oper701	103824	127584	M1	1
C219	Oper782	127584	129824	M1	1
C35	Oper709	129824	144800	M1	1
C43	Oper1318	144800	146908	M1	1
C249	Oper716	146908	163036	M1	1
C234	Oper724	163036	180956	M1	1
C239	Oper686	180956	194396	M1	1
C87	Oper1119	194396	200444	M1	1
C37	Oper812	200444	209984	M1	1
C194	Oper107	209984	210734	M1	1
C237	Oper916	210734	213590	M1	1
C169	Oper731	213590	219990	M1	1
C20	Oper1453	219990	221046	M1	1
C187	Oper748	221046	236046	M1	1
C176	Oper926	236046	240486	M1	1
C268	Oper493	240486	241356	M1	1
C129	Oper97	241356	242988	M1	1
C57	Oper841	242988	249132	M1	1
C205	Oper210	249132	264162	M1	1
C89	Oper1125	264162	268166	M1	1
C151	Oper1453	268166	272742	M1	1
C228	Oper196	272742	281582	M1	1
C273	Oper147	281582	300302	M1	1
C207	Oper701	300302	308222	M1	1
C93	Oper282	308222	314722	M1	1
C270	Oper10	314722	330490	M1	1
C241	Oper177	330490	332730	M1	1
C200	Oper1853	332730	336402	M1	1
C38	Oper679	336402	343122	M1	1
C211	Oper841	343122	350162	M1	1
C15	Oper1453	350162	354738	M1	1

Utilaj	Utilizare	Inceput	Sfarsit
M1	1	0	583191
M2	1	0	286202
M5	1	0	552484
M6	1	0	526786
M7	1	0	589611
M8	1	0	579115
M9	1	0	518098
M10	1	0	429442
M11	1	5166	422775
M13	1	285222	288342
M14	1	25757	532324
M15	1	36	768424
M16	1	54	1091321
M17	1	90	999668
M18	1	0	1091705
M19	1	318	1090901
M20	1	11016	1083515
M22	1	11529	631041
M23	1	11289	1087045
M24	1	11799	1085537
M25	1	292971	293181
M26	1	12279	973396
M27	1	20265	1023612
M29	1	598389	1059324
M30	1	319581	1078284
M31	1	11499	637989
M32	1	366	1089409
M33	1	12464	1061484
M34	1	11115	1091501
M38	1	44859	731147
M43	1	100227	1086035
M44	1	982	1091801
M45	1	24140	807377
M48	1	10992	971260

Figura 3.9. Captură de ecran privind modul în care datele de ieşire au fost verificate in Excell după implementarea celui de al doilea script Visual Basic pentru verificarea condiției de nesuprapunere a două operațiuni în același timp pe aceiași mașină.

(sursa: prelucrare proprie)

Interpretare și perspective:

1. **Valoarea obiectivului**: Valoarea obiectivului de 1.091.828 reprezintă timpul de finalizare, adică timpul necesar pentru a finaliza ultima operațiune. Această valoare indică cât de bine minimizează programul timpul total de producție.
2. **Eficiența capacității de soluționare**: Abilitatea acesteia de a explora 9,3 milioane de ramuri și de a găsi o soluție optimă în aproximativ 13 secunde demonstrează eficiența CPLEX în gestionarea problemelor complexe de programare.
3. **Constrângeri și variabile**: Numărul mare de constrângeri (4.997) și variabile de decizie (2.224) subliniază complexitatea problemei, inclusiv necesitatea gestionării mai multor comenzi, operațiuni și constrângeri de precedență.
4. **Non-suprapunere și precedență**: Asigurarea non-suprapunerii operațiunilor pe același utilaj și respectarea constrângerilor de precedență sunt critice pentru crearea unui program fezabil și eficient.

Indicatori cheie ai programării cu Cplex Constrain Programing în OPL

1. **Utilizarea variabilelor de interval și secvență**:
    - **Variabile de interval**: Variabila intOperatiuni este definită ca o variabilă de decizie de interval care reprezintă momentele de început și sfârșit ale operațiunilor. Aceasta este o caracteristică centrală a programării cu constrângeri, fiind foarte potrivită pentru probleme de programare.
    - **Variabile de secvență**: Variabila conflicte este o variabilă de decizie de secvență folosită pentru gestionarea conflictelor pe utilaje, asigurând că nicio două operațiuni nu se suprapun pe același utilaj.
2. **Gestionarea constrângerilor**:
    - **Constrângeri de non-suprapunere**: Utilizarea constrângerii noOverlap este o tehnică tipică de programare cu constrângeri pentru a asigura că operațiunile pe același utilaj nu se suprapun în timp.
    - **Constrângeri de precedență**: Constrângerile precum endBeforeStart asigură că operațiunile sunt programate în ordinea corectă, un alt aspect definitoriu al programării cu constrângeri.
    - **Constrângeri de utilizare**: Aceste constrângeri definesc momentele de început și sfârșit ale utilizării utilajelor, gestionate eficient prin programare cu constrângeri.
3. **Funcția obiectiv**:

- Funcția obiectiv de minimizare a timpului maxim de finalizare (minimize max) este esențială pentru optimizarea procesului de producție, reducând timpul total necesar pentru finalizarea tuturor operațiunilor.

Modelul CPLEX gestionează eficient o problemă complexă de programare cu mai multe constrângeri și variabile. Utilizează funcții avansate, cum ar fi variabilele de interval și secvență, pentru a modela problema în mod natural și eficient. Statisticile soluției și performanța solverului indică faptul că CPLEX este bine adaptat pentru astfel de sarcini de optimizare, oferind soluții robuste și scalabile. Putem spune că problema este rezolvată folosind o abordare de "programare cu constrângeri" (CP) în OPL (Optimization Programming Language). Modelul și implementarea sa în CPLEX, așa cum sunt văzute în fișierele și capturile de ecran furnizate, demonstrează caracteristicile tipice ale unei abordări de programare cu constrângeri. [121]

Acest model realizat in CPLEX OPL a fost gândit să poată fi aplicat în orice situație similară, adică poate fi aplicat și în alte cazuri în care problema programării producției are date de intrare de tip similar. Un anumit număr de comenzi i , pentru un anumit număr de produse k ce pot fi prelucrat pe un anumit număr de mașini m, fiecare produs având propriul itinerar tehnologic. În concluzie, modelul Guinet, [47] care a fost identificat ca soluție optimă pentru problema de programare a producției, a fost implementat cu succes, oferind rezultate remarcabile. Prin utilizarea metodei de rezolvare oferite de CPLEX Constraint Programming, funcția obiectiv a fost optimizată astfel încât să genereze o valoare inferioară timpului maxim permis pentru livrarea produselor, realizând un timp de rulare de doar 12,55 secunde)

În lumea dinamică a producției industriale, optimizarea proceselor de fabricație este esențială pentru a asigura eficiența, reducerea costurilor și îmbunătățirea productivității. Programarea producției implică alocarea optimă a resurselor, gestionarea ordinii operațiunilor și respectarea termenelor limită, toate acestea într-un cadru caracterizat de constrângeri variate și complexe. Un instrument valoros pentru abordarea acestor probleme este Programarea cu Constrângeri (CP), implementată în platforme precum IBM ILOG CPLEX Optimization Studio. Modelul dezvoltat în CPLEX OPL, despre care discutăm aici, a fost conceput pentru a fi aplicabil în diverse scenarii similare, oferind o soluție flexibilă și robustă pentru multiple tipuri de probleme de programare a producției.

Acest model a fost proiectat cu scopul de a fi aplicabil într-o gamă largă de situații în care problema programării producției prezintă date de intrare de tip similar. Concret, modelul poate gestiona un anumit număr de comenzi \(i\), pentru un anumit număr de produse \(k\), care pot fi prelucrate pe un anumit număr de mașini \(m\). Fiecare produs are un itinerar tehnologic propriu,

adică o succesiune de operațiuni care trebuie efectuate într-o anumită ordine pe diferite utilaje și într-un interval de timp specificat.

Modelul dezvoltat în CPLEX OPL se remarcă prin flexibilitatea sa, putând fi aplicat în diverse scenarii de programare a producției care prezintă caracteristici similare. Această adaptabilitate este esențială în mediul industrial actual, unde cerințele și condițiile se pot schimba rapid. Mai jos sunt detaliate câteva exemple de situații în care acest model poate fi aplicat eficient.

În producția în serii mici și medii, diversitatea produselor și variabilitatea cererii impun necesitatea unui sistem de programare flexibil și adaptabil. Modelul CP din CPLEX OPL poate gestiona eficient comenzile pentru diferite produse, fiecare având propriul itinerar tehnologic, asigurând utilizarea optimă a utilajelor și respectarea termenelor de livrare.

În producția de masă, unde procesele sunt adesea foarte standardizate, dar cu itinerare tehnologice complexe, modelul poate optimiza alocarea resurselor și secvențele de operațiuni pentru a minimiza timpul de producție și a reduce costurile.

În industriile auto și aeronautică, programarea producției implică adesea gestionarea unor itinerare tehnologice extrem de complexe și multiple constrângeri de precedență. Modelul CP poate asigura respectarea riguroasă a acestor constrângeri, optimizând în același timp utilizarea resurselor și reducând timpii de inactivitate a utilajelor.

Industria electronică și a dispozitivelor medicale necesită adesea procese de producție extrem de precise și bine coordonate. Modelul poate gestiona cu succes aceste cerințe, optimizând procesele de fabricație pentru a asigura calitatea produselor și respectarea termenelor de livrare.

Avantajele utilizării modelului Guinet în CPLEX OPL sunt:

- Modelul permite optimizarea utilizării resurselor și a ciclurilor de producție, ceea ce conduce la creșterea eficienței și productivității. Prin minimizarea timpilor de inactivitate a utilajelor și asigurarea unui flux continuu al operațiunilor, companiile pot produce mai mult într-un timp mai scurt.
- Capacitatea modelului de a se adapta la diverse scenarii de producție îl face un instrument valoros pentru companiile care trebuie să răspundă rapid la schimbările cererii și condițiilor de piață. Modelul poate fi ajustat cu ușurință pentru a gestiona noi comenzi, produse și itinerare tehnologice.
- prin optimizarea proceselor de producție, modelul contribuie la reducerea costurilor operaționale. Utilizarea eficientă a resurselor și reducerea ciclurilor de producție duc la economii semnificative în ceea ce privește consumul de energie, materii prime și forță de muncă.
- un alt avantaj al utilizării modelului CP (Constrain Programming) este îmbunătățirea calității produselor. Prin asigurarea respectării stricte a itinerarelor tehnologice și a constrângerilor de

producție, modelul contribuie la menținerea standardelor de calitate și la reducerea defectelor de fabricație.

Modelul realizat în CPLEX OPL reprezintă o soluție puternică și flexibilă pentru problemele complexe de programare a producției. Capacitatea sa de a gestiona un număr variabil de comenzi, produse și utilaje, fiecare cu propriile itinerare tehnologice, îl face aplicabil într-o gamă largă de industrii și scenarii. Prin optimizarea utilizării resurselor și a ciclurilor de producție, modelul contribuie la creșterea eficienței, reducerea costurilor și îmbunătățirea calității produselor. Astfel, acest model se dovedește a fi un instrument esențial pentru companiile care doresc să își îmbunătățească procesele de producție și să rămână competitive pe piața globală.

Pe de altă parte, abordarea prin Programare cu Constrângeri, tot prin intermediul CPLEX OPL, a demonstrat o capacitate mult mai mare de a gestiona complexitatea problemei. Această metodă a reușit să ofere o soluție care s-a încadrat în termenii contractuali, respectând limitele impuse de timp, după cum se poate vedea în figura 3.8 și Figura 3.9. Aceasta indică faptul că metodele bazate pe constrângeri pot fi mai eficiente pentru problemele de optimizare cu multiple condiții de frontieră, oferind o flexibilitate sporită în modelarea problemelor complexe.

Tabel 3.7. Rezultate Cplex Constrain Programming

Cplex Constrain Programming	Timp de productie contractual					Timp de productie obtinut		
	Sapt.	Zile	Ore 2*16	Min.	Sec.	Ore	Sec.	Timp rulare
	6	30	480	28800	1728000	303.3	1091828	12.55

(sursa: prelucrare proprie)

Datele folosite au fost preluate din fișierul Date_EF, Sheet 3, fișier în care acestea au fost aranjate în așa fel încât modelul folosit să le poate prelua. Această aranjare a datelor a fost realizată cu ajutorului unui script Visual Basic prezentat în Anexa 4.

Rezultatele obținute au fost exportate în fișierul Date_EF, Sheet 4, fișier în care acestea au fost exprimate cu o exactitate la nivel de secundă pentru a fi siguri că nu există suprapuneri, (prezentat în figura următoare).

Am procedat astfel la verificarea uneia din cele mai importante condiții ale programului și anume aceea prin care pe aceiași mașină sa nu se suprapună două operații în același timp. Acest lucru a fost realizat cu ajutorul unui al doilea script Visual Basic care a realizat automat această verificare care poate fi văzută in fisierul Date_EF, Sheet6, prezentate în Anexa 5.

Capturile de ecran din anexe demonstrează modul în care CPLEX OPL CP gestionează problema de optimizare, oferind statistici detaliate și valoarea obiectivului rezultat. Informațiile cheie din capturile de ecran pot fi rezumate astfel (a se vedea următoarele două figuri):

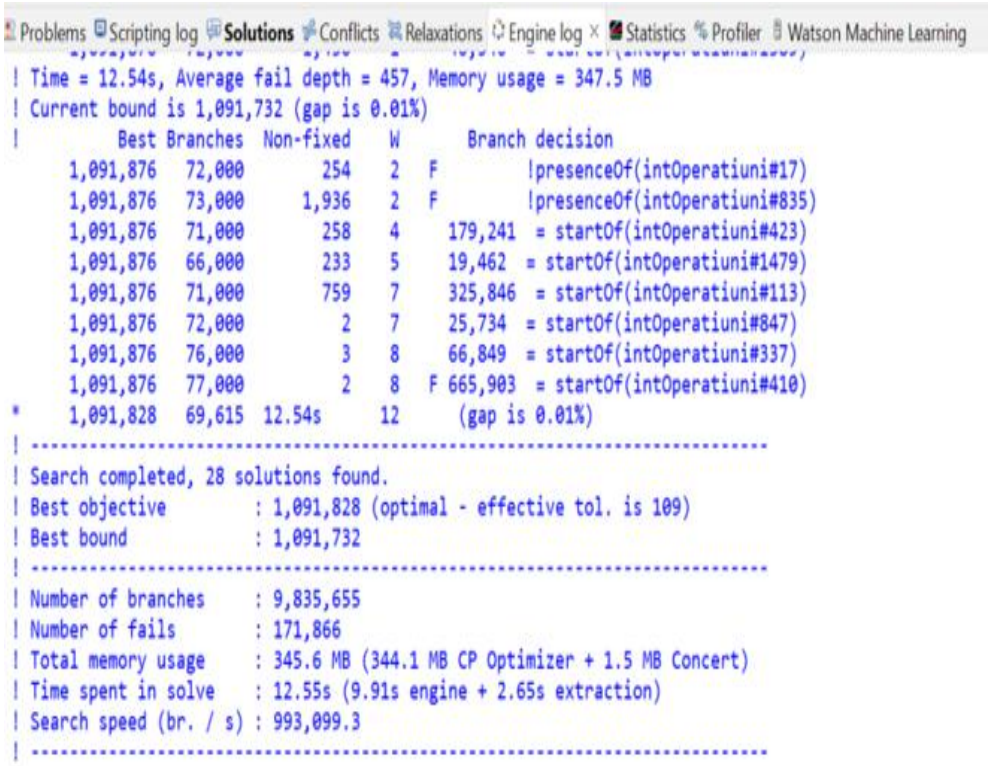
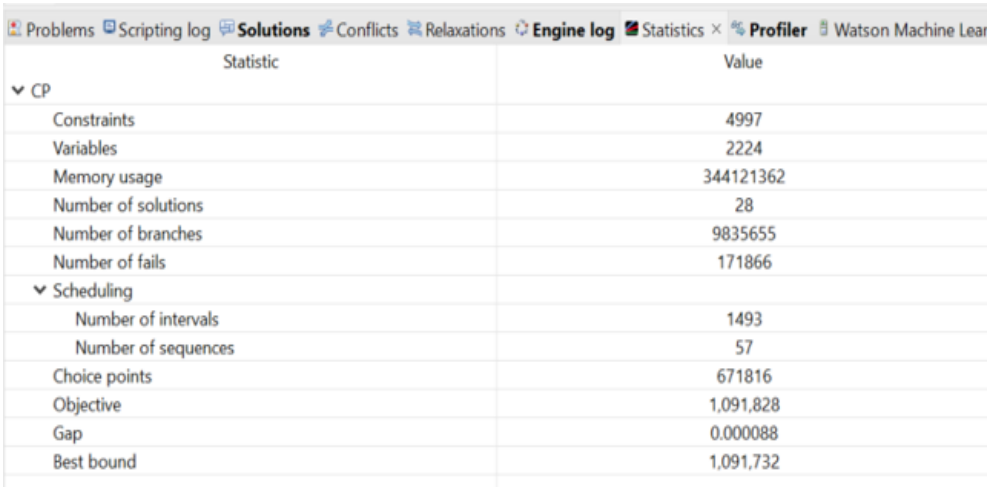


Figura 3.10. Valorile Engine Log după rularea modelului CP in CPLEX OPL (sursa: prelucrare proprie)



Statistic	Value
CP	
Constraints	4997
Variables	2224
Memory usage	344121362
Number of solutions	28
Number of branches	9835655
Number of fails	171866
Scheduling	
Number of intervals	1493
Number of sequences	57
Choice points	671816
Objective	1,091,828
Gap	0.000088
Best bound	1,091,732

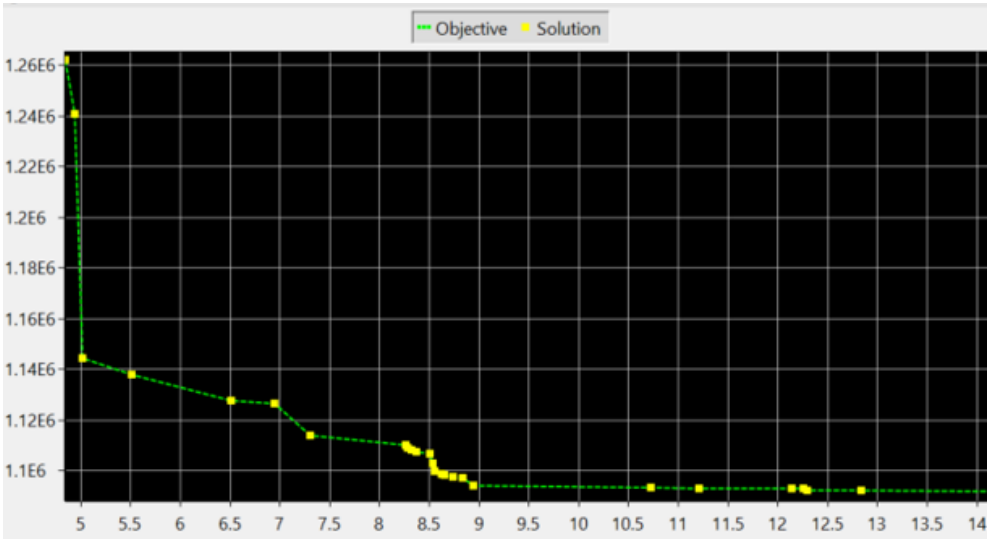


Figura 3.11. Statistica rulării modelului CP în CPLEX OPL
(sursa: prelucrare proprie)

În concluzie, modelul Guinet, care a fost identificat ca soluție optimă pentru problema de programare a producției, a fost implementat cu succes, oferind rezultate remarcabile. Prin utilizarea metodei de rezolvare oferite de CPLEX Constraint Programming, funcția obiectiv a fost optimizată astfel încât să genereze o valoare inferioară timpului maxim permis pentru livrarea produselor. Această performanță indică nu doar eficiența modelului Guinet, ci și capacitatea superioară a CPLEX Constrain Programing de a gestiona probleme complexe de programare a producției. Conform datelor prezentate în tabelul nr. 3.7. de mai sus, timpul de rulare pentru găsirea soluției optime a fost redus semnificativ, ajungând la doar 12,55 secunde. Acest timp de procesare extrem de scurt demonstrează eficiența computațională a metodei CPLEX Constraint Programming în comparație cu alte metode tradiționale. Prin urmare, integrarea modelului Guinet în cadrul acestui context metodologic a permis nu doar optimizarea procesului de producție, dar și asigurarea respectării termenelor stricte de livrare, consolidând astfel performanța operațională a companiei. Această realizare subliniază relevanța și aplicabilitatea practică a modelului Guinet în domeniul programării producției, precum și eficiența soluțiilor oferite de CPLEX Constrain Programing în contextul industrial modern.

3.4.2.2. Metode euristice

3.4.2.2.1. Algoritm Genetic (AG) în Matlab

Algoritmul Genetic (*en. Genetic Algorithm*- AG) reprezintă o abordare evoluționistă de optimizare, fundamental diferită de tehnica de programare liniară utilizată în CPLEX. În timp ce

programarea liniară se bazează pe metode deterministe pentru a rezolva ecuații și inecuații liniare, AG folosește principii inspirate din biologia naturală, cum ar fi selecția, încrucișarea și mutația, pentru a evolua soluții la probleme complexe.

În cadrul unui algoritm genetic, fiecare cromozom reprezintă o posibilă soluție la problemă. Acesta poate fi vizualizat ca o matrice, în care rândurile corespund numărului de comenzi, iar coloanele reprezintă intervalele orare. Dimensiunea populației alese determină numărul total al acestor matrice. Pentru a înțelege fluxul algoritmului genetic, fișierul initialization.m oferă o bază fundamentală. În această etapă de inițializare, fiecare cromozom este plasat aleatoriu în limitele sale, respectând constrângerile definite pentru ora de început și ora de sfârșit, specificate clar în fișierul de inițializare. Aptitudinea fiecărui cromozom, adică timpul total de producție necesar pentru finalizarea tuturor comenzilor, este apoi evaluată. [120]

Un aspect crucial al AG este că, odată ce comenzile sunt limitate în cadrul inițializării, aceste constrângeri nu vor fi încălcate în buclele ulterioare ale algoritmului. În bucla principală a AG, selecția părinților se face pe baza aptitudinii lor. Părinții selecționați participă la un proces de încrucișare, combinându-și genele pentru a genera descendenți. Acești descendenți pot fi ulterior supuși unui proces de mutație, unde anumite gene sunt alterate aleatoriu pentru a introduce variabilitate în populație. Noua populație este evaluată și clasată pe baza aptitudinii, continuând ciclul iterativ până la atingerea unui criteriu de oprire, cum ar fi un număr predefinit de generații sau o valoare țintă a fucției obiectiv.

Această metodă de optimizare, deși diferită de abordarea liniară, oferă flexibilitate și adaptabilitate. Algoritmul genetic permite explorarea unui spațiu mare de soluții posibile și este capabil să găsească soluții bune într-un timp rezonabil. În comparație cu tehnicile deterministe, care pot fi limitate de complexitatea și dimensiunea problemei, AG se dovedește a fi o soluție robustă pentru optimizarea problemelor complexe de producție.

Implementarea AG nu numai că facilitează adaptarea la cerințele fluctuante ale pieței, dar și optimizează utilizarea resurselor, reducând riscurile asociate cu prognozele pe termen lung. În concluzie, deși abordările liniară și genetică diferă semnificativ, combinarea acestora poate oferi soluții mai eficiente și adaptabile pentru optimizarea proceselor de producție.

Algoritmul Genetic propus precum și lista detaliată de parametrii și variabile a acestui algoritm prezentat în detaliu în Anexa 6 și Anexa 7, este:

1. Cromozomi:

- **Semnificație**: Reprezintă o posibilă soluție în Algoritmul Genetic (AG).
- **Scop**: Structurat ca o matrice cu rânduri egale cu numărul de comenzi și coloane reprezentând numărul de ore.

2. Definirea fazei inițiale:

- **orderStart**: Tablou cu orele de început pentru comenzi.
    - **Scop**: Definește cele mai timpurii ore posibile de început pentru comenzi.
- **orderEnd**: Tablou cu orele de sfârșit pentru comenzi.
    - **Scop**: Definește cele mai târzii ore posibile de sfârșit pentru comenzi.
- **latestStart**: Tablou cu cele mai târzii ore de început pentru comenzi.
    - **Scop**: Asigură că comenzile sunt programate în fereastra de producție permisă.
- **productionTime**: Totalul orelor necesare pentru producția fiecărei comenzi.
    - **Scop**: Calculează timpul de producție pentru fiecare comandă pe baza datelor furnizate.

3. Bucla principala a Algoritmului Genetic:

- **Population**: Matrice 3D care reprezintă populația curentă de cromozomi.
    - **Scop**: Stochează multiple soluții posibile.
- **Objective Function**: Calculează timpul total de producție pentru fiecare cromozom.
    - **Scop**: Evaluează cât de bine îndeplinește fiecare cromozom obiectivul de a minimiza timpul de producție.
- **Selection**: Selectează soluțiile parentale pe baza valorii fucției obiectiv
    - **Scop**: Alege cei mai performanți cromozomi pentru a produce următoarea generație.
- **Crossover**: Combină perechi de soluții parentale pentru a produce descendenți.
    - **Scop**: Creează noi cromozomi prin combinarea genelor soluțiilor parentale.
- **Mutation**: Introduce schimbări aleatorii în descendenți.
    - **Scop**: Asigură diversitatea genetică în populație prin modificarea aleatorie a genelor.

4. Variabile principale:

- **orderIdx**: Indexul comenzii curente.
    - **Scop**: Urmărește poziția fiecărei comenzi în buclă.
- **machineMatrix**: Matrice care indică disponibilitatea mașinilor pentru fiecare produs.
    - **Scop**: Asigură că mașinile sunt alocate corect produselor.
- **truecol**: Coloanele corespunzătoare mașinilor disponibile pentru produsul comenzii curente.
    - **Scop**: Identifică ce mașini sunt disponibile pentru programarea comenzii curente.
- **prod_num**: matrice de numere de produse pentru fiecare comandă.
    - **Scop**: Mapează fiecare comandă la produsul corespunzător.
- **current individual index in population**: Indexul individului curent în populație.
    - **Scop**: Urmărește poziția fiecărui cromozom în populație.

Acești parametri și variabile sunt esențiali pentru funcționarea corectă a unui Algoritm Genetic, permițând evaluarea și optimizarea soluțiilor pentru a îmbunătăți performanța proceselor de producție.

Tabel 3.8. Parametrii și variabile pentru funcționarea algoritmului genetic

Nr	PROBLEME	DENUMIRE	FIȘIER	LINIE	EXPLICAȚIE (DACĂ ESTE CAZUL)
1	$f = min \sum_{i=1}^{I} T_{in,M_p}$	Fucția obiectiv	**operation.m**	line 46	**totalProductionTime(j,1) = sum(any(population(:,:,j), 1));**
2	$T_{in,m_p} - T_{jn,m_p} + KZ_{in,jn,m_p} \geq d_{jn,m_p}$	Constrangere nr. 1	**machinechk.m**	line 4 to line 24	Ajută la identificarea sloturilor de mașini active pentru a preveni suprapunerile operațiilor conform Ecuațiilor 2 și 3
3	$T_{jn,m_p} - T_{in,m_p} + K\left(1 - Z_{in,jn,m_p}\right) \geq d_{in,m_p}$	Constrangere nr. 2	**machinechk.m**	line 4 to line 24	Ajută la identificarea sloturilor de mașini active pentru a preveni suprapunerile de lucrări conform Ecuațiilor 2 și 3. Asistă în planificare prin identificarea coloanelor utilizate, relevante pentru gestionarea programării secvențiale a lucrărilor conform Ecuației 4.
4	$T_{in,m_p+1} - T_{in,m_p} \geq d_{in,m_p}$	Constrangere nr. 3	**machinechk.m**	line 4 to line 24	Evită orice suprapunere a aceleași operații pe diferite mașini și asigură că fluxul de lucru este logic și secvențial.
5	$T_{in,1} \geq dint_i$	Constrangere nr.4	**initializePopulation .m**	43-108	Aceasta este limita pe care o vom stabili în etapa de inițializare. Consultați variabila cu numele de început.
6	$T_{in,m_P} \leq dl_i - d_{in,m_P}$	Limitare nr. 5	**initializePopulation .m**	43-108	Nici o operație nu ar trebui să înceapă înainte de momentul stabilit. Consultați variabilele cu numele de început.
7	Timpul de producție pentru fiecare comandă, de la primire, este de 6 săptămâni (6 * 5 * 16 ore = 480 ore).	Condiție 1	**initializePopulation .m**	43-108	Inițializat de la început, iar tot codul din această fereastră este respectat, așa cum se vede în fișierul menționat.
8	Comenzile finalizate trebuie să respecte coloana *Data scadentă a comenzii de lucru* (WO Due Date).	Condiție 1	**initializePopulation .m**	43-108	În mod similar, această constrângere este respectată în etapa de inițializare. Ulterior, restricțiile sunt respectate în funcțiile AG.

(sursa: prelucrare proprie)

Constrângerea nr. 1, 2 și 3 (machinechk.m, lines 4 to 24) necesită următoarele mențiuni:

 Aceste linii asigură că programarea comenzii curente nu se suprapune cu sloturile active ale comenzilor anterioare. Aceasta corespunde direct constrângerii prin verificarea disponibilității mașinii și menținerea secvențierii corecte, prevenind astfel conflictele de programare.

Rezumat:

 Liniile 4-10: Identifică comenzile anterioare care folosesc aceeași mașină.

 Liniile 11-24: Găsește și combină sloturile active pentru aceste comenzi pentru a asigura că nu există suprapuneri.

Constrângerea nr. 4 și 5 (initializePopulation.m, lines 43 to 108) necesită următoarele mențiuni:

 Liniile 43-46: Extrag și calculează condițiile limită pentru orele de început.

 Liniile 47-50: Se asigură că orele de început sunt în fereastra disponibilă.

 Liniile 51-108: Validarea și aplicarea condițiilor limită în timpul inițializării populației.

Condițiile 1 și 2 (initializePopulation.m, lines 43 to 108) sunt identice și anume:

 Aceste linii asigură că timpii de producție pentru fiecare lucrare sunt în intervalul permis. Rezumat:

 Liniile 43-46: Extrag și calculează condițiile limită pentru orele de început și timpii de producție.

 Liniile 47-50: Se asigură că orele de început sunt în fereastra disponibilă.

 Liniile 51-108: Validarea și aplicarea condițiilor limită în timpul inițializării populației.

Pentru mai multe detalii privindmodul în care funcționează liniile de cod, a se vedea ANEXA 5.

Algoritmii Genetici, implementați în Matlab [123], au oferit rezultate mixte. Deși aceștia au rulat într-un timp relativ scurt, nu au reușit să livreze o soluție care să se încadreze în timpul contractual de șase săptămâni. Cea mai mică valoare a funcției obiectiv obținută prin rularea acestui tip de Algoritm Genetic se regăsește în următoarea captură de ecran.

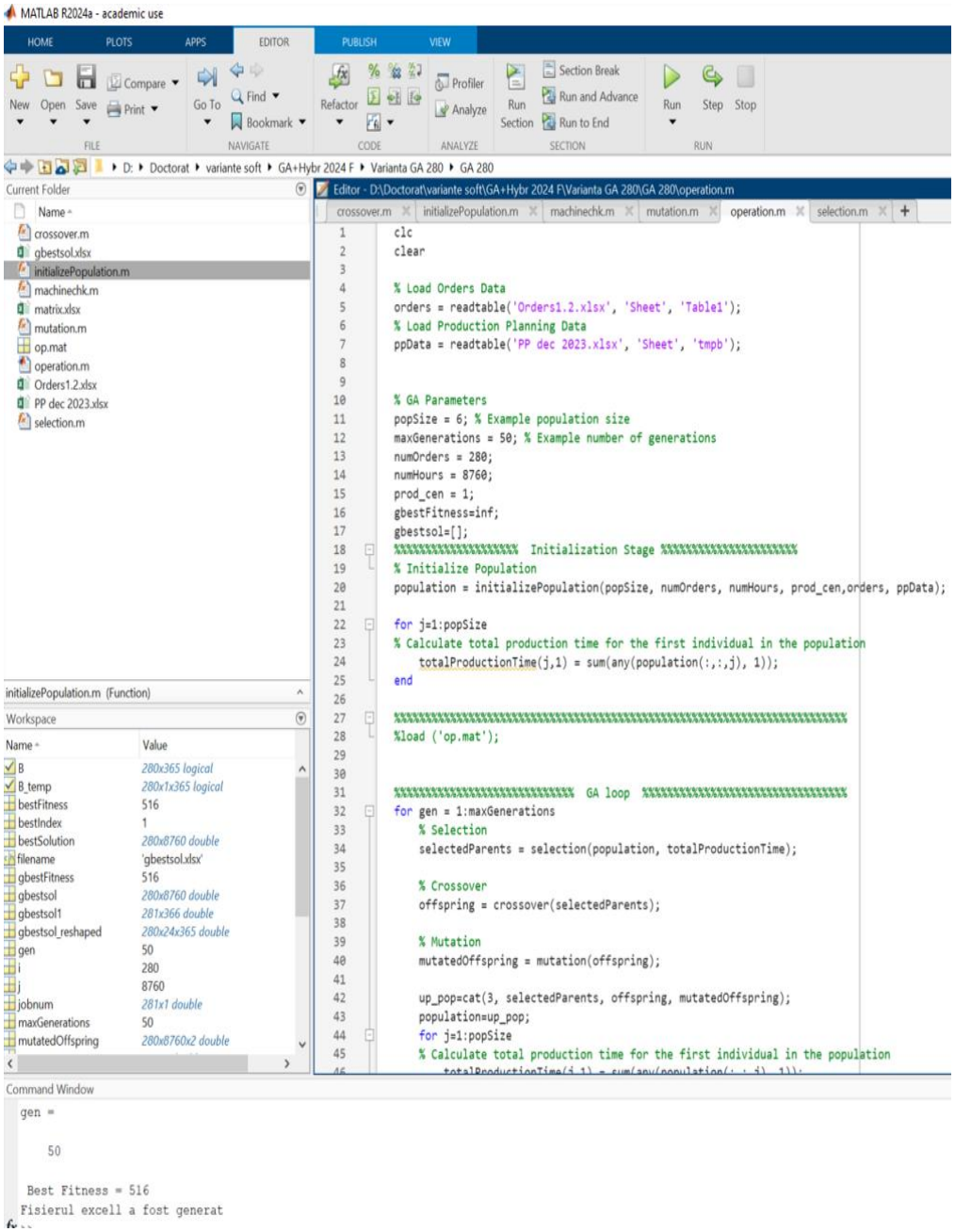


Figura 3.12. Captură de ecran privind valoarea minimă a Funcției Obiectiv obținută după rularea AG (sursa: prelucrare proprie)

Aceasta sugerează că, deși Algoritmii Genetici sunt puternici în explorarea spațiului de căutare și pot găsi soluții bune în probleme complexe, performanța lor poate fi limitată de natura

funcției obiectiv sau de modul în care sunt setate constrângerile. În acest context, ajustarea fină a parametrilor algoritmului, cum ar fi rata de mutație și rata de crossover, ar putea îmbunătăți performanța, dar ar putea fi necesar și un model combinat pentru a atinge obiectivele impuse.

Tabel 3.9. Rezultate Matlab AG

Matlab AG	Timp de productie contractual					Timp de productie obtinut		
	Saptamani	Zile	Ore 2*16	Minute	Secunde	Ore	Secunde	Timp rulare
	6	30	480	28800	1728000	516	1857600	21

(sursa: prelucrare proprie)

3.4.2.2.2. Algoritm Hybrid (AG+ PSO) în Matlab

Bucla principală a unui algoritm hibrid de optimizare care combină Particle Swarm Optimization (PSO) și Genetic Algorithm (AG) este esențială pentru obținerea unei soluții eficiente și optimizate și este detaliată în Anexa 8. Această buclă constă în cinci pași principali: 1) actualizarea vitezei și poziției particulelor (PSO), 2) evaluarea fucției obiectiv a populației, 3) operarea AG (selecție, crossover și mutație), 4) combinarea și selecția celor mai buni indivizi, și 5) recalcularea funcție obiectiv pentru populația combinată. În continuare, vom detalia fiecare dintre acești pași pentru a înțelege modul în care funcționează și cum interacționează PSO și AG pentru a obține cele mai bune rezultate.

1) Primul pas în bucla principală este actualizarea vitezei și poziției particulelor, care reprezintă soluțiile candidate în algoritmul PSO. Fiecare particulă din populație își actualizează viteza și poziția pe baza experiențelor sale anterioare și a celor mai bune soluții găsite de întregul grup (swarm). Viteza fiecărei particule este influențată de trei componente: componenta de inerție, componenta cognitivă și componenta socială. Componenta de inerție menține particula în mișcare în direcția actuală, contribuind la explorarea spațiului soluțiilor. Componenta cognitivă reprezintă experiența individuală a particulei și tinde să o readucă spre cea mai bună poziție personală găsită până acum. Componenta socială reflectă influența colectivă a întregului grup și îndrumă particula către cea mai bună poziție globală găsită de swarm. Aceste trei componente sunt ponderate și combinate pentru a determina noua viteză a particulei. După calcularea noii viteze, poziția particulei este actualizată prin adăugarea vitezei la poziția curentă, permițându-le să exploreze spațiul soluțiilor și să se apropie de soluția optimă.

2) Odată ce pozițiile particulelor au fost actualizate, următorul pas este evaluarea funcției obiectiv pentru fiecare particulă. Funcția obiectiv este o măsură a calității unei soluții candidate și este definită în funcție de problema specifică abordată. Pentru fiecare particulă din populație, se calculează funcția obiectiv pe baza noii sale poziții, evaluând soluția propusă de particulă și determinând cât de bine îndeplinește aceasta criteriile de performanță stabilite. După evaluarea funcția obiectiv, particulele își actualizează cea mai bună poziție personală (pBest) dacă soluția curentă este mai bună decât cea mai bună soluție găsită anterior. În plus, dacă una dintre particule a găsit o soluție care este mai bună decât cea mai bună soluție globală (gBest) cunoscută până acum, atunci se actualizează și cea mai bună poziție globală. Acest proces de evaluare și actualizare a funcției obiectiv este crucial pentru ghidarea particulelor către soluțiile optime și pentru asigurarea convergenței algoritmului către cea mai bună soluție posibilă.

3) AG este integrat în bucla principală pentru a adăuga variație și diversitate în populație. AG introduce trei operatori principali: selecția, crossover-ul și mutarea. Selecția se face de obicei pe baza funcției obiectiv, favorizând indivizii cu un rezultat mai bun. O metodă comună de selecție este turneul, în care un număr de indivizi sunt selectați aleatoriu și cel mai bun dintre aceștia este ales ca părinte.

Selecția prin turneu asigură că soluțiile mai bune au o șansă mai mare de a fi selectate, dar menține și diversitatea prin includerea unor indivizi mai slabi. După selecție, crossover-ul combină genele a doi părinți pentru a produce descendenți. Punctul de trecere (en. crossover) este ales aleatoriu și genele sunt schimbate între cei doi părinți la acel punct. Acest proces permite combinarea trăsăturilor bune de la doi părinți diferiți, generând soluții potențial mai bune. Mutarea introduce mici variații aleatorii în descendenți pentru a menține diversitatea genetică și pentru a preveni convergența prematură. Prin aplicarea acestor operatori genetici, AG contribuie la explorarea și exploatarea spațiului soluțiilor, completând procesul de optimizare realizat de PSO.

4) După aplicarea operatorilor genetici, descendenții generați de AG sunt combinați cu populația actualizată de PSO. Extinderea populației: Descendenții generați de AG sunt adăugați la populația existentă, creând o populație mai mare care conține toate soluțiile posibile din acea generație. Selecția celor mai buni indivizi: Din această populație extinsă, se selectează cei mai buni indivizi pentru a forma noua populație pentru următoarea generație. Selecția este bazată pe funcția obiectiv, asigurându-se că cele mai bune soluții sunt păstrate și propagate. Această combinare și selecție asigură că populația evoluează în direcția soluțiilor optime, menținând în același timp diversitatea necesară pentru a evita convergența prematură.

5) În ultima etapă a buclei principale, funcția obiectiv a populației combinate este recalculată pentru a actualiza pBest și gBest. Reevaluarea funcției obiectiv.: Fiecare individ din populația combinată este reevaluat pentru a determina funcția obiectiv în contextul noilor condiții. Acest lucru poate implica recalcularea timpului total de producție sau a altor criterii de performanță relevante. Acest proces de reevaluare asigură că toate soluțiile sunt reevaluate în contextul noilor descendenți și al modificărilor introduse de AG, asigurând convergența către cea mai bună soluție posibilă..

În contextul algoritmilor genetici hibrizi, un aspect crucial este procesul de actualizare a valorilor pBest și gBest. Acest proces intervine după recalcularea funcției obiectiv, actualizând valoarea pBest pentru fiecare particulă dacă funcția obiectiv curent este mai bună decât cea anterioară. De asemenea, gBest este actualizat dacă există o soluție care îmbunătățește cea mai bună soluție globală cunoscută. Prin acest mecanism, algoritmul asigură o comparație și evaluare corectă a tuturor soluțiilor, păstrând focusul pe identificarea celor mai bune opțiuni posibile.

Interacțiunea dintre Particle Swarm Optimization (PSO) și Genetic Algorithm (AG) în cadrul buclei principale este esențială pentru eficiența și succesul unui algoritm hibrid. Fiecare metodă aduce contribuții specifice și complementare la procesul global de optimizare.

PSO se ocupă de explorarea spațiului soluțiilor prin actualizarea pozițiilor particulelor bazate pe experiențele individuale și colective. Este recunoscut pentru eficiența sa în găsirea rapidă a soluțiilor datorită mecanismului său de învățare colectivă. Totuși, PSO poate suferi de convergență prematură către soluții locale, limitând astfel explorarea spațiului soluțiilor.

AG, pe de altă parte, introduce variație și diversitate prin operatorii săi genetici – selecția, crossover-ul și mutarea. Acești operatori asigură că populația rămâne diversă și că noi soluții sunt constant explorate. AG contribuie astfel la evitarea convergenței premature și menține un nivel ridicat de explorare a spațiului soluțiilor.

Bucla principală a unui algoritm hibrid de optimizare care combină PSO și AG este un proces iterativ complex. Acesta implică actualizarea pozițiilor și vitezelor particulelor, evaluarea funcției obiectiv, aplicarea operatorilor genetici și selecția celor mai buni indivizi. Fiecare pas este esențial pentru asigurarea convergenței algoritmului către soluții optime. Interacțiunea dintre PSO și AG permite exploatarea rapidă a soluțiilor bune și menținerea diversității populației, prevenind astfel convergența prematură și asigurând o explorare completă a spațiului soluțiilor. Această combinație sinergică maximizează șansele de a găsi soluții eficiente și optimizate pentru probleme complexe, cum ar fi programare a producției.

Funcția de selecție este un element esențial în cadrul algoritmilor genetici, având rolul crucial de a alege indivizii care vor contribui la formarea generațiilor viitoare. Algoritmul propus prezintă o implementare hibridă care îmbină algoritmul de optimizare prin roi de particule (PSO) cu metodele genetice pentru a selecta indivizii eficient. Această funcție de selecție utilizând PSO îmbunătățește calitatea soluțiilor generate prin intermediul unui proces evolutiv iterativ.

Principiile funcției de selecție folosind PSO. Selecția în cadrul algoritmului genetic este inspirată de principiile selecției naturale, unde indivizii cu trăsături favorabile au șanse mai mari de a-și transmite genele descendenților. În cadrul funcției de selecție utilizând PSO, acest principiu este integrat cu mecanismele de optimizare specifice PSO, implicând actualizarea pozițiilor și vitezelor particulelor pentru a identifica soluțiile optime.

PSO este un algoritm de optimizare inspirat de comportamentul colectiv al roiurilor de păsări sau al bancurilor de pești. Fiecare particulă reprezintă o soluție posibilă, iar pozițiile și vitezele particulelor sunt actualizate în funcție de experiența proprie și de cea a întregului roi. Funcția de selecție utilizând PSO în algoritmul genetic implică selectarea părinților pentru reproducere, evaluarea funcției obiectiv și aplicarea operatorilor genetici pentru a genera noi descendenți.

Procesul de selecție. Funcția de selecție utilizând PSO se desfășoară în mai mulți pași esențiali, contribuind la formarea unei populații de indivizi cu performanțe superioare. Evaluarea funcției obiectiv a fiecărui individ este primul pas, urmat de selecția prin turneu, unde se selectează un grup de indivizi din populație și se compară funcția obiectiv a acestora. Cel mai bun individ din acest grup este ales pentru reproducere. Integrarea PSO în selecție introduce un mecanism suplimentar de optimizare, actualizând pozițiile și vitezele particulelor după selecția inițială a părinților. Astfel, fiecare particulă este influențată de cea mai bună poziție personală (pBest) și de cea mai bună poziție globală (gBest) identificată de întregul roi. Această actualizare continuă permite explorarea eficientă a spațiului soluțiilor și îmbunătățirea continuă a funcției obiectiv

Generarea descendenților. După selecția părinților și actualizarea pozițiilor și vitezelor particulelor prin PSO, următorul pas este generarea descendenților prin aplicarea operatorilor genetici: crossover și mutare. Crossover-ul implică combinarea genelor a doi părinți pentru a crea descendenți, iar mutarea introduce variații aleatorii pentru a menține diversitatea genetică și a preveni convergența prematură.

Combinația și selecția celor mai buni indivizi. După generarea descendenților, aceștia sunt combinați cu populația actuală pentru a forma o populație extinsă. Din această populație extinsă, se

selectează cei mai buni indivizi pe baza funcției obiectiv pentru a forma noua populație pentru următoarea generație. Populația extinsă include atât indivizii actualizați prin PSO, cât și descendenții generați prin AG, asigurând păstrarea și propagarea celor mai bune soluții și menținând diversitatea genetică necesară pentru explorarea spațiului soluțiilor. Acest lucru este explicat in cele ce urmează:

1. Dimensiunea Turneului (tournamentSize=5)

Scop: Dimensiunea turneului se referă la numărul de indivizi care sunt selectați aleatoriu din populație pentru a concura într-un turneu. Câștigătorul turneului, adică cel mai bine adaptat individ din grup, este selectat pentru a contribui la generarea următoarei generații.

Avantaje ale setării tournamentSize=5:

- **Promovarea diversității genetice:** O dimensiune a turneului de 5 este considerată moderată. Aceasta permite includerea unor indivizi mai puțin adaptați în procesul de selecție, ceea ce ajută la menținerea diversității genetice în populație. Diversitatea genetică este esențială pentru a evita convergența prematură, unde populația ar putea converge rapid la o soluție suboptimă și nu ar explora alte potențiale soluții optime.
- **Echilibru între exploatare și explorare:** TournamentSize=5 oferă un echilibru între exploatarea soluțiilor actuale bune și explorarea de noi regiuni în spațiul de soluții. Într-o dimensiune a turneului prea mică, riscul de a selecta indivizi mai puțin adaptați crește, ceea ce poate încetini convergența. Pe de altă parte, o dimensiune prea mare favorizează prea mult indivizii cei mai adaptați, ceea ce poate reduce diversitatea și crește riscul de convergență prematură.
- **Rezistența la fluctuații:** Cu tournamentSize=5, algoritmul genetic devine mai robust în fața fluctuațiilor în funcția obiectiv a indivizilor, asigurându-se că selecția nu este excesiv de sensibilă la mici variații de funcție obiectiv. Acest lucru poate contribui la o evoluție mai stabilă a populației, menținând în același timp capacitatea de a găsi soluții optime pe termen lung.

2. Numărul de Părinți (numParents=20)

Scop: Numărul de părinți se referă la câți indivizi sunt selectați din populația curentă pentru a contribui la generarea urmașilor în generația următoare. Aceștia sunt utilizați în procesele de încrucișare și mutație pentru a crea noi indivizi.

Avantaje ale setării numParents=20:

- **Diversitate genetică ridicată:** Alegerea a 20 de părinți pentru reproducere într-o populație tipică este o setare relativ ridicată, ceea ce favorizează menținerea unei diversități genetice considerabile în cadrul populației. Aceasta înseamnă că mai multe combinații genetice pot fi explorate, crescând șansa de a descoperi soluții mai bune și de a evita stagnarea evoluției.
- **Presiune de selecție moderată:** Cu un număr mare de părinți, presiunea de selecție este mai moderată, ceea ce înseamnă că algoritmul are șanse mai mari de a păstra trăsături genetice utile dintr-o generație la alta. Deși acest lucru poate încetini convergența, avantajul constă în crearea unei populații mai robuste, capabile să exploreze mai eficient spațiul de căutare.
- **Flexibilitate și adaptabilitate:** Un număr mai mare de părinți poate fi avantajos în probleme complexe, unde spațiul de căutare este vast și variabil. În astfel de cazuri, menținerea unei game largi de trăsături genetice permite populației să se adapteze mai bine la schimbările de mediu sau la variațiile funcției obiectiv.

În concluzie, setările `tournamentSize=5` și `numParents=20` oferă avantaje semnificative în contextul unui algoritm genetic. Dimensiunea turneului de 5 asigură un echilibru sănătos între exploatarea soluțiilor bune existente și explorarea unor noi soluții potențial mai bune, contribuind la evitarea convergenței premature. De asemenea, selectarea a 20 de părinți promovează diversitatea genetică și menține o presiune de selecție moderată, facilitând o explorare mai amplă a spațiului de soluții și creând o populație mai adaptabilă și robustă. Aceste setări sunt deosebit de utile în probleme complexe, unde menținerea diversității și explorarea continuă sunt critice pentru succesul pe termen lung al algoritmului.

Recalcularea funcției obiectiv**.** În ultima etapă a funcției de selecție, funcția obiectiv a populației combinate este recalculată pentru a actualiza pBest și gBest. Acest proces implică reevaluarea fiecărui individ pentru a determina funcția obiectiv în noile condiții, actualizând astfel cele mai bune poziții personale și globale. După recalcularea funcției obiectiv, pBest și gBest sunt actualizate pentru a reflecta cele mai bune soluții globale cunoscute. Acest proces asigură o comparație și evaluare corectă a tuturor soluțiilor, menținând focalizarea pe găsirea celor mai bune soluții posibile.

Lista detaliata a parametrilor si variabilelor pentru Algoritmul Hybrid AG+PSO pentru studiul de caz (programarea productiei la ABC)

1. Cromozomi:

Semnificație: Reprezintă o posibilă soluție în Algoritmul Genetic (AG).

Scop: Structurat ca o matrice cu rânduri egale cu numărul de comenzi și coloane reprezentând numărul de ore.

2. Definirea Fazei Initiale:

orderStart: Tablou cu orele de început pentru comenzi.

Scop: Definește cele mai timpurii ore posibile de început pentru comenzi.

orderEnd: Tablou cu orele de sfârșit pentru comenzi.

Scop: Definește cele mai târzii ore posibile de sfârșit pentru comenzi.

latestStart: Tablou cu cele mai târzii ore de început pentru comenzi.

Scop: Asigură că comenzile sunt programate în fereastra de producție permisă.

productionTime: Totalul orelor necesare pentru producția fiecărei comenzi.

Componentele PSO

Viteza: reprezintă vectorul care definește direcția și mărimea deplasării fiecărei particule dintr-un roi (swarm) în spațiul de căutare.

Scop: Defineste mișcarea particulei intre pBest si gBest

Personal Best (pBest): este cea mai bună poziție individuală a particulei

Scop: Definește punctul de referință în mișcarea particulei

Global Best (gBest): este cea mai bună poziție globală descoperită de roi.

Scop: Direcționează întregul roi spre poziția optimă

Componentele AG

Population: Matricea 3D a populației curente a cromozomilor.

Scop: Conține mai multe posibile soluții.

Objectiv funcție: Calculează valoarea minimă a funcției obiectiv.

Scop: Evaluează dacă cromozomii răspund cerinței funcției obiectiv.

Selecția: Selectează părinții pe baza scorurilor utilizând o strategie de selecție turneu îmbunătățită de informațiile din PSO (Particle Swarm Optimization).

Scop: Alege cei mai buni cromozomi pentru a deveni părinți ai următoarei generații de descendenți.

Încrucișarea: Combină părinți pentru a produce descendenți printr-o abordare procentuală.

Scop: Creează noi descendenți încrucișând genele părinților, permițând explorarea unor noi zone în care se găsesc soluțiile.

Mutația: Introduce modificări aleatorii în rândul descendenților.

Scop: Asigură diversitatea genetică a populației.

Integrarea AG cu PSO

Mecanismul Hybrid: Oscilează între actualizările PSO pentru viteză și poziție și AG pentru selecție, încrucișare și mutație.

Scop: Exploatează punctele forte ale ambilor algoritmi. Exploatarea pentru AG și explorarea pentru PSO, în scopul îmbunătățirii capacității algoritmului de a găsi soluții mai bune și de a evita soluțiile de optim local.

Variabile principale

gen: actuala generație folosită în bucla principală a algoritmului.

Scop: Monitorizează progresul algoritmului prin generațiile sale.

totalProductionTime: O matrice care conține timpul total de producție pentru fiecare cromozom.

Scop: Este folosită pentru a calcula timpul total de producție.

machineMatrix: Matrice care indică disponibilitatea mașinilor pentru fiecare produs. **Scop:** Asigură că mașinile sunt alocate corect produselor.

Operațiuni PSO

Buclele principale gestionează evoluția populației de soluții folosind atât tehnici PSO (Optimizarea prin Roiuri de Particule), cât și AG (Algoritmi Genetici).

Linia 51-52

r1 = rand(size(population(:, :, i)));

r2 = rand(size(population(:, :, i)));

Factorii aleatori „r1” și „r2” sunt generați pentru fiecare particulă, influențând actualizarea vitezei pe baza celor mai bune soluții personale și globale.

Linia 53-55

velocity(:, :, i) = w * velocity(:, :, i) + c1 * r1 .* (pBest(:, :, i) - population(:, :, i)) ...+ c2 * r2 .* (gBest - population(:, :, i));

Viteza fiecărei particule este actualizată. Noua viteză este calculată luând în considerare viteza anterioară și diferența dintre poziția curentă și atât cea mai bună poziție personală, cât și cea globală, ponderate de parametrii PSO „w”, „c1” și „c2”.

Linia 56

population(:, :, i) = population(:, :, i) + velocity(:, :, i);

Poziția fiecărei particule este actualizată prin adăugarea vitezei nou calculate. Acest pas deplasează particula către soluții potențial mai bune pe baza vitezei actualizate.

Linia 59

% population(:, :, i) = max(min(population(:, :, i), upperBound), lowerBound);

Această linie comentată sugerează o abordare pentru a asigura că pozițiile actualizate nu depășesc limitele predefinite, deși nu este activă în implementarea curentă. Asigurarea limitelor valide poate fi crucială pentru menținerea soluțiilor fezabile.

Operațiuni ale Algoritmului Genetic (AG**)**. După actualizarea particulelor folosind PSO, componentele AG sunt aplicate pentru a optimiza în continuare populația. Aceste componente includ selecția, încrucișarea și mutația, care ajută la menținerea diversității și explorarea de noi soluții.

Selecția este realizată folosind o strategie de selecție prin turneu, așa cum este descris în funcția „selection.m”. Acest proces implică selectarea celor mai buni indivizi pe baza scorurilor funcției obiectiv dintr-un subset ales aleatoriu al populației. Integrarea cu rezultatele PSO implică potențial selecția între cele mai bune soluții PSO și candidații AG, asigurând păstrarea soluțiilor de înaltă calitate.

Încrucișarea este detaliată în funcția „crossover.m”, unde doi părinți sunt aleși pentru a produce urmași prin combinarea informațiilor genetice. Acest lucru este realizat prin împărțirea soluțiilor părinților la un punct de încrucișare definit și schimbarea segmentelor lor pentru a crea noi urmași. Această metodă permite algoritmului să moștenească trăsături benefice de la ambii părinți, conducând potențial la soluții mai bune.

Mutația, așa cum este implementată în funcția „mutation.m”, introduce modificări aleatorii la urmași, ceea ce ajută la evitarea optimelor locale și creșterea diversității genetice. Procesul de mutație implică alterarea aleatorie a genelor sub o rată controlată, asigurând că modificările sunt benefice, dar nu prea disruptive.

Soluții și vizualizare

gbestsol: Este valoarea soluției celei mai bune obținute. **Scop:** Folosită pentru a programa producția. **Visualization:** Vizualizarea grafică a programării producției (Diagrama Gantt). **Scop:** Ajută la înțelegerea distribuției comenzilor ce urmează a intra în producție în funcție de timp și itinerariul tehnologic.

Abordarea hibridă, care combină Algoritmii Genetici cu Optimizarea prin Roiuri de Particule (PSO), a demonstrat o performanță îmbunătățită față de utilizarea Algoritmilor Genetici în mod izolat. În cadrul experimentelor, în trei cazuri din șaizeci, algoritmii hibrizi au reușit să furnizeze soluții care

respectă condiția de încadrare în intervalul de timp pentru livrarea produselor oferind cicluri de producție de 352, 446 și 480 ore de producție. În tabelul 3.10. este redat cel mai bun timp obținut.

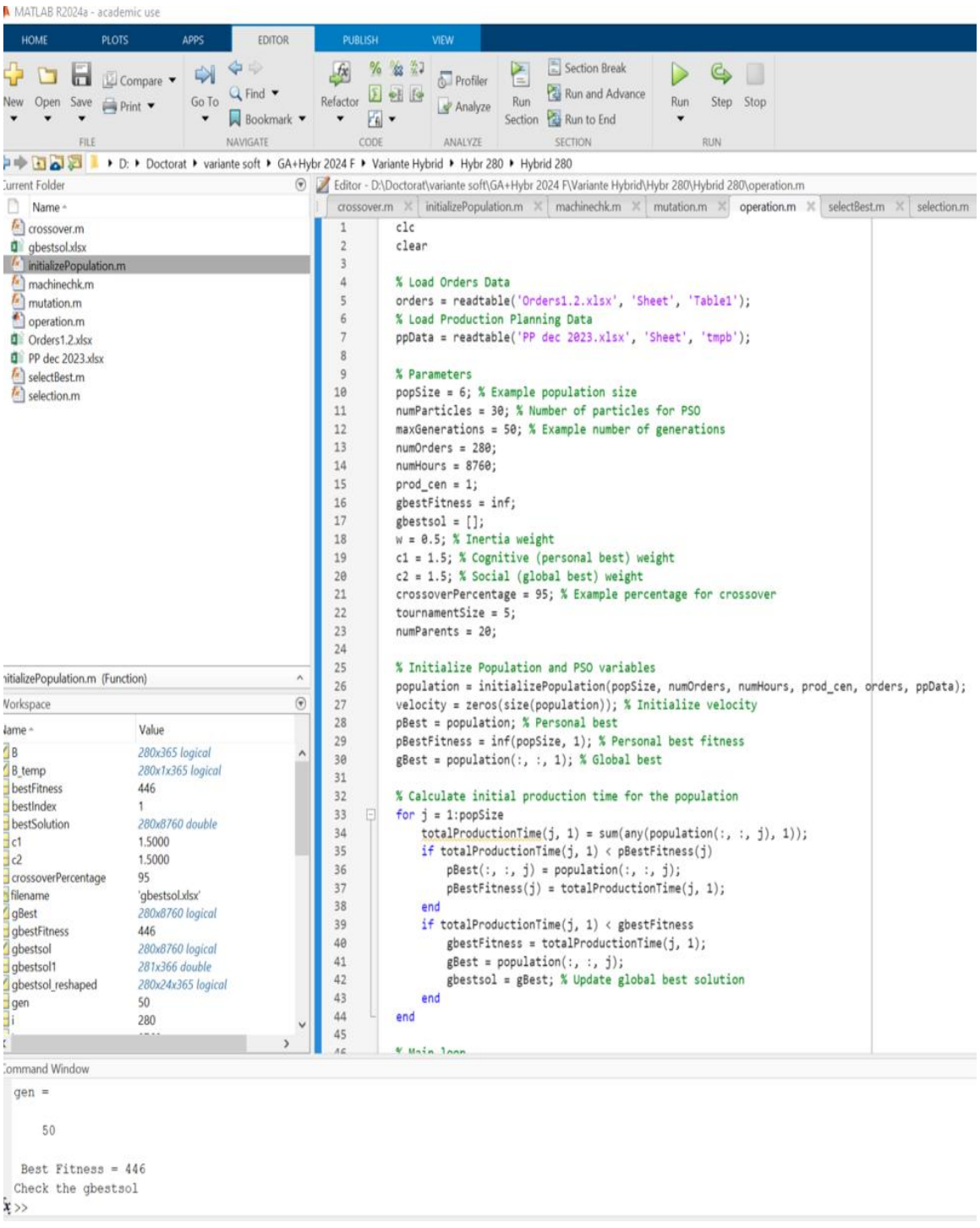
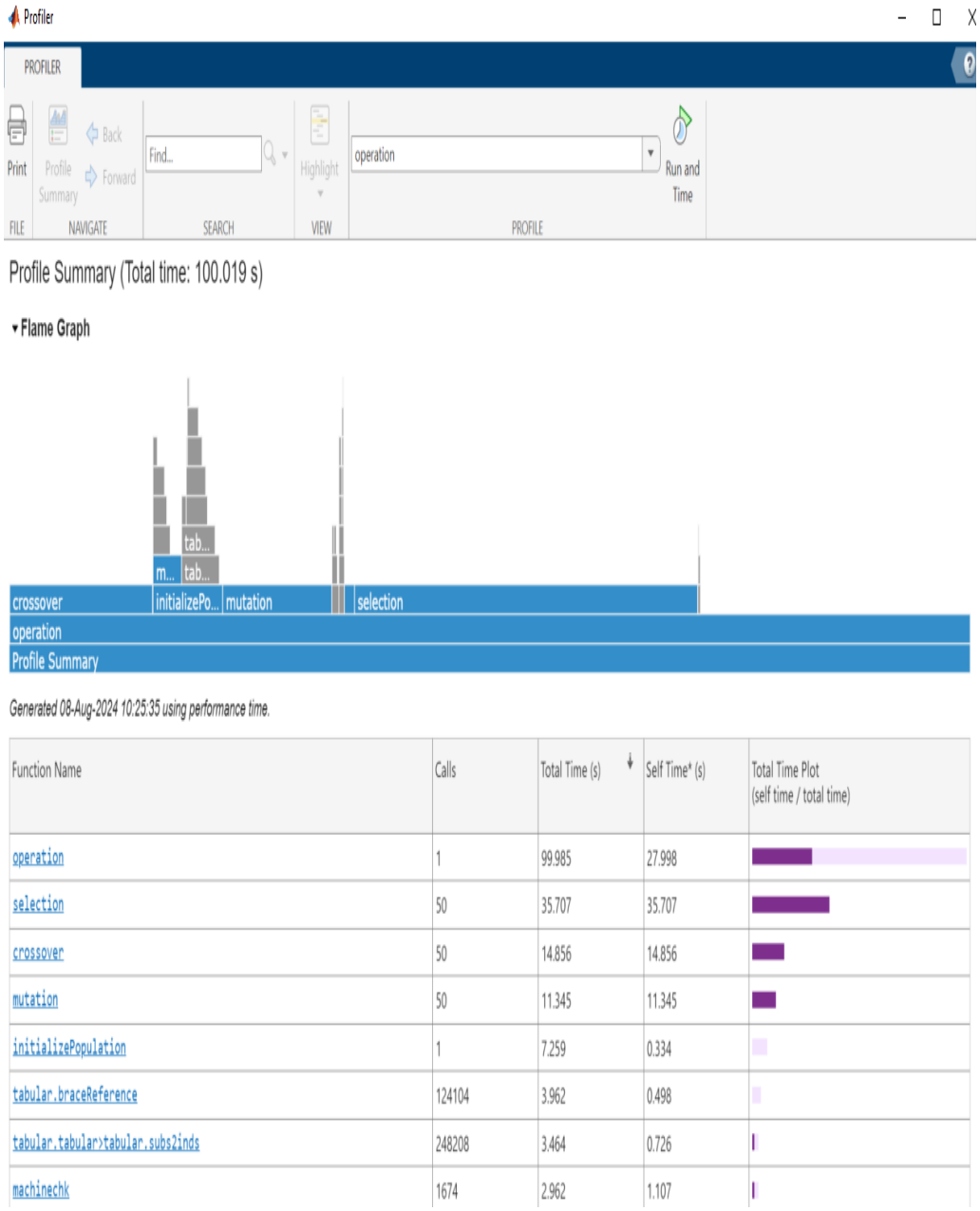


Figura 3.13. Captură de ecran privind valoarea minimă a funcției obiectiv obținută cu ajutorul Algoritmului Hybrid (AG+PSO)
(sursa: prelucrare proprie)



Function Name	Calls	Total Time (s)	Self Time* (s)	Total Time Plot (self time / total time)
operation	1	99.985	27.998	
selection	50	35.707	35.707	
crossover	50	14.856	14.856	
mutation	50	11.345	11.345	
initializePopulation	1	7.259	0.334	
tabular.braceReference	124104	3.962	0.498	
tabular.tabular>tabular.subs2inds	248208	3.464	0.726	
machinechk	1674	2.962	1.107	

Figura 3.14. Captură de ecran privind timpul de rulare al Algoritmului Hybrid (AG+PSO) (sursa: prelucrare proprie)

Configurarea principalilor parametrii ai Algoritmului Hybrid (AG+PSO) pentru care a fost obținută cea mai bună valoare a funcției obiectiv (352 ore) este detaliată în captura de ecran anterioară, iar timpul de rulare a indicat urmărele (conform figurii anterioare):

- Factorul de mutație : 0,7%
- Populație AG : 50
- Factorul de încrucișare : 95%
- Numărul de particule PSO : 30

- Factorul Inerției PSO w : 0,5
- Coeficientul cognitiv c1 : 1,5
- Coeficientul social c2 : 1,5

Aceaste rezultate sugerează că integrarea a două metode de optimizare poate oferi beneficii semnificative, exploatând punctele forte ale fiecărei metode pentru a depăși limitările individuale. În acest caz, PSO a contribuit la îmbunătățirea exploatării în spațiul de căutare, însă nu a fost suficient pentru a respecta complet termenii contractuali de timp, indicând faptul că, deși abordările hibride pot oferi soluții mai bune, acestea necesită încă o ajustare fină și o analiză aprofundată a interacțiunii dintre metode pentru a maximiza eficiența.

Tabel 3.10. Rezultate Matlab Hybrid (AG+PSO)

Matlab Hybrid(Ag+PSO)	Timp de productie contractual					Timp de productie obtinut		
	Sapt.	Zile	Ore 2*16	Minute	Secunde	Ore	Secunde	Timp rulare
	6	30	480	28800	1728000	352	1267200	111

(sursa: prelucrare proprie)

CAPITOLUL 4. ANALIZA REZULTATELOR OBȚINUTE

4.1. Rezultate. Discuții

Compararea metodelor exacte și și a celor euristice:

A) În privința preciziei și optimizării:

- Metodele exacte, cum ar fi MILP și CP, sunt capabile să găsească soluții optime pentru probleme bine definite și cu constrângeri clare. Aceste metode sunt foarte precise și garantează găsirea celei mai bune soluții posibile. Cu toate acestea, ele pot fi ineficiente pentru probleme foarte mari sau foarte complexe din punct de vedere computațional.
- Pe de altă parte, metodele euristice, cum ar fi AG și algoritmii hibrizi, nu garantează găsirea soluției optime, dar pot găsi soluții foarte bune într-un timp rezonabil. Aceste metode sunt mai flexibile și pot aborda probleme pentru care metodele exacte ar fi impracticabile din cauza complexității sau dimensiunii mari a spațiului soluțiilor.

B) În privința flexibilității și adaptabilității:

- Metodele exacte sunt mai puțin flexibile deoarece necesită o formulare matematică strictă a problemei. Orice schimbare în problemă necesită o reformulare completă a modelului matematic. În contrast, metodele euristice sunt mult mai flexibile și pot adapta ușor la schimbările din parametrii problemei sau din mediul de operare.

C) În privința complexității implementării:

- Metodele exacte pot fi complexe de implementat și necesită cunoștințe avansate de matematică și programare. Formularea corectă a problemei și setarea corespunzătoare a parametrilor sunt esențiale pentru succesul acestor metode. Pe de altă parte, metodele euristice sunt, în general, mai ușor de implementat și de adaptat la diferite probleme. Algoritmii genetici și hibrizii pot fi implementați folosind librării de cod existente și pot fi ajustați ușor pentru a îmbunătăți performanța.

D) În privința performanței computaționale:

- Metodele exacte pot deveni foarte lente pentru problemele de dimensiuni mari din cauza necesității de a explora exhaustiv spațiul soluțiilor. Aceasta poate duce la timpi de calcul foarte mari și la necesitatea de resurse computaționale semnificative. Metodele euristice, deși pot fi mai rapide, necesită adesea multiple iterații și pot consuma, de asemenea, resurse computaționale considerabile, dar într-un mod mai gestionabil și scalabil.

E) În privința capacității de a oferi soluții:

- Metodele exacte oferă soluții foarte precise și deterministe, dar pot fi sensibile la schimbări minore în datele de intrare sau în formularea problemei. Metodele euristice sunt mai robuste la variațiile din date și pot găsi soluții bune chiar și atunci când datele sunt incomplete sau incorecte.

Atât metodele exacte cât și cele euristice au punctele lor forte și slabe în contextul planificării producției. Alegerea metodei potrivite depinde de natura specifică a problemei, de resursele disponibile și de cerințele de precizie și eficiență. Metodele exacte sunt ideale pentru probleme bine definite și unde se cere precizie maximă, în timp ce metodele euristice sunt mai potrivite pentru probleme complexe, unde flexibilitatea și adaptabilitatea sunt esențiale. Utilizarea combinată a ambelor tipuri de metode, în funcție de context, poate oferi soluțiile cele mai bune pentru optimizarea planificării producției.

Procesul de rezolvare a funcției obiectiv, care a fost formulată prin metoda Guinet, a inclus implementarea a patru abordări distincte. Fiecare dintre aceste abordări a fost aleasă în funcție de capacitatea sa de a aborda complexitatea problemei și de a livra rezultate optime în cadrul constrângerilor impuse. Prima abordare a fost utilizarea Programării Liniare, implementată prin CPLEX OPL, o metodă tradițională, dar eficientă în probleme de optimizare, cu toate că prezintă limitări în contextul unor funcții obiectiv cu multiple constrângeri. A doua abordare, Programarea cu Constrângeri, tot prin intermediul CPLEX OPL, a oferit o flexibilitate sporită în modelarea problemelor și a permis integrarea mai ușoară a constrângerilor complexe. Rezolvarea modelului Guinet adaptat, prin metoda CP, reprezentată de funcția obiectiv însoțită de constrângerile și condițiile aferente, precum și ordonarea eficientă a activităților de producție a rezolvat problema studiului de caz, prin eliminarea locurilor înguste prezentate în Tabelul 3.1

În continuare, a fost evaluată utilizarea Algoritmilor Genetici (AG), implementați în Matlab, cunoscuți pentru capacitatea lor de a explora eficient spațiul de căutare și de a găsi soluții apropiate de optim în probleme complexe, însă cu provocări în ceea ce privește convergența rapidă către soluții optime într-un interval de timp restrâns. În final, o abordare hibridă între Algoritmii Genetici și Optimizarea prin Roiuri de Particule (PSO), de asemenea implementată în Matlab, a fost testată pentru a evalua potențialul acestei combinații de a îmbunătăți atât explorarea, cât și exploatarea spațiului de soluții, maximizând astfel șansele de a găsi soluția optimă în limitele stabilite.

Fiecare dintre aceste abordări a fost evaluată riguros în funcție de capacitatea sa de a satisface constrângerile impuse și de a minimiza timpul de producție în limitele stabilite. În urma acestor

analize, s-a evidențiat faptul că alegerea metodei optime depinde de specificul problemei și de capacitatea fiecărei metode de a integra și rezolva constrângerile complexe ale procesului de producție. Aceasta subliniază necesitatea unei abordări flexibile și adaptabile în procesul de optimizare a producției, în special în contextul unor industrii dinamice și competitive.

4.2. Analiza cauză-efect

Analiza cauză-efect este esențială în identificarea relațiilor dintre diversele aspecte operaționale și rezultatele obținute, oferind un cadru clar pentru evaluarea problemelor și soluțiilor în procesele de producție. În acest caz, s-a optat pentru reprezentarea sub formă de tabel în locul unei diagrame Ishikawa, deoarece complexitatea variabilelor analizate impune o structurare clară și

concisă a informațiilor. Tabelul permite evidențierea relațiilor directe între cauze și efecte, oferind o prezentare detaliată și precisă. Spre deosebire de diagrama Ishikawa, care se concentrează pe identificarea generală a cauzelor, tabelul facilitează compararea rapidă a soluțiilor și a rezultatelor, fiind mai eficient în contextul unei analize detaliate și tehnice.

Tabel 4.1. Analiza cauză-efect

CAUZE	EFECTE
1. Complexitatea ridicată a proceselor de producție	**1. Necesitatea algoritmilor de optimizare avansați.** Complexitatea ridicată a proceselor de producție de serie, care implică numeroase etape de fabricație, variabile și constrângeri, necesită utilizarea algoritmilor de optimizare avansați. Algoritmii genetici (AG), optimizarea prin roiuri de particule (PSO) și metodele hibride (AG + PSO) sunt esențiale pentru a aborda aceste probleme complexe. Acești algoritmi permit explorarea unui spațiu vast de soluții posibile și identificarea celor mai eficiente configurații pentru liniile de producție, maximizând astfel eficiența și reducând costurile operaționale. În acest moment al cercetării noastre inconstanța soluțiilor oferite ne fac să credem ca această idee este o viitoare direcție de cercetare.
2. Variabilitatea cererii pieței	**2. Flexibilitatea și adaptabilitatea în planificarea producției.** Variabilitatea cererii pieței impune necesitatea unei flexibilități și adaptabilități sporite în planificarea producției. Algoritmii de optimizare trebuie să fie capabili să răspundă rapid la schimbările cererii, ajustând parametrii de producție și alocarea resurselor. De exemplu, utilizarea tehnicilor de optimizare precum CP permite ajustarea dinamică a programelor de producție pentru a răspunde prompt la fluctuațiile cererii, asigurând astfel continuitatea și eficiența operațiunilor.
3. Costurile ridicate de stocare	**3. Minimizarea stocurilor prin optimizarea resurselor.** Costurile ridicate asociate gestionării stocului determină necesitatea minimizării stocurilor prin optimizarea resurselor. Algoritmii de optimizare, cum ar fi MILP și CP, sunt utilizați

	pentru a determina cantitățile optime de producție și stocare, reducând astfel costurile asociate. Aceste metode permit o planificare eficientă a producției, asigurând că doar cantitățile necesare sunt produse și stocate, contribuind astfel la reducerea costurilor de stocare și îmbunătățirea cash flow-ului.
4. Limitările de timp și resurse	**4. Eficientizarea programării și alocării resurselor.** Limitările de timp și resurse necesită o programare și alocare eficientă a acestora pentru a maximiza productivitatea. Algoritmii precum AG+PSO și CP permit optimizarea secvențelor de operații și alocarea resurselor, asigurând utilizarea optimă a capacităților de producție. Acești algoritmi ajută la reducerea timpilor de așteptare și la creșterea eficienței operaționale, contribuind la respectarea termenelor de livrare și la utilizarea eficientă a resurselor disponibile, oferind timpi de rulare la nivelul câtorva secunde.
5. Necesitatea reducerii costurilor de producție	**5. Implementarea metodelor de optimizare a costurilor.** Pentru a rămâne competitive, companiile trebuie să reducă costurile de producție. Implementarea metodelor de optimizare, cum ar fi CP și AG+PSO, permite identificarea celor mai eficiente soluții pentru reducerea costurilor (cu speranța că în viitor algoritmii hibrizi (AG+PSO) vor putea oferi rezultate mai concludente). Aceste metode ajută la optimizarea utilizării materialelor și a resurselor, reducând deșeurile și costurile de producție. Astfel, companiile pot oferi produse la prețuri competitive fără a compromite calitatea.
6. Necesitatea îmbunătățirii calității produselor	**6. Utilizarea IA pentru controlul calității.** Îmbunătățirea calității produselor este esențială pentru satisfacția clienților și competitivitatea pe piață. Tehnicile avansate de IA, cum ar fi deep learning și reinforcement learning, pot fi utilizate pentru controlul calității și detectarea defectelor în timp real. Aceste metode permit monitorizarea continuă a proceselor de producție și ajustarea parametrii pentru a menține standardele de calitate,

	reducând astfel rata de defecte și costurile asociate reparațiilor și retururilor.
7. Necesitatea predictibilității în ceea ce privește procesul de mentenanță	**7. Reducerea timpilor de inactivitate și a costurilor de întreținere.** Mentenanța predictivă este crucială pentru reducerea timpilor de inactivitate și a costurilor de întreținere. Utilizarea IoT și a algoritmilor de optimizare permite monitorizarea continuă a echipamentelor și anticiparea defecțiunilor. Acest lucru asigură intervenții prompte și eficiente, prevenind avariile costisitoare și asigurând continuitatea operațiunilor. Astfel, companiile pot menține echipamentele în stare optimă de funcționare, reducând costurile de întreținere și maximizând productivitatea.
8. Necesitatea inovării continue	**8. Stimularea dezvoltării și adoptării de tehnologii noi.** Inovația continuă este esențială pentru menținerea competitivității și adaptarea la schimbările rapide din industrie. Companiile trebuie să investească în cercetare și dezvoltare pentru a adopta și implementa tehnologii noi, cum ar fi AI și big data, în procesele de producție. Acest lucru permite îmbunătățirea continuă a eficienței operaționale și adaptabilității, contribuind la dezvoltarea de produse inovatoare și la creșterea satisfacției clienților. Adopția de tehnologii noi stimulează inovația.
9. Evoluția internă a algoritmilor genetici și hibrizi.	**9. Inconstanța rezultatelor oferite**, din cauza specificului evoluției acestor tipuri de algoritmi. Din 50 de rulări a algoritmilor hibrizi, cu diverși parametrii, cu diverse setări ale parametrilor au fost obținute doar 3 rezutate poziționate în intervalul corect (0-480 de ore), dar și acelea erau diferite între ele.
10. Contanța metodelor exacte	**10. Constrain Programming** oferă cele mai bune soluții în acest moment pentru că oferă același rezultat bun, stabil, sigur, extrem de detaliat. A fost obținută cea mai mică valoare a funcției obiectiv și anume 303 ore, valoare constantă indiferent de numărul de rulări a modelului.

(sursa: prelucrare proprie)

4.3. Evaluarea rezultatelor prin analiza SWOT

Tabel 4.2. Analiza S.W.O.T. a rezultatelor cercetării.

PUNCTE TARI	PUNCTE SLABE
1. Precizia și rigoarea metodelor matematice exacte. Utilizarea metodelor exacte precum programarea liniară mixtă (MILP) și programarea cu constrângeri (CP) permite obținerea unor soluții extrem de precise și riguroase. Aceste metode sunt capabile să	1. **Complexitatea și timpul de calcul al metodelor exacte** - deși precise, pot fi extrem de complexe și pot necesita un timp de calcul semnificativ pentru probleme mari și complexe. De exemplu, programarea liniară în CPLEX poate deveni impracticabilă pentru

furnizeze soluții optime pentru probleme bine definite și să asigure consistența rezultatelor indiferent de numărul de rulări. De exemplu, utilizarea CPLEX pentru programarea liniară garantează că soluțiile obținute sunt exacte și replicabile, aspect esențial pentru planificarea strategică în producția de serie. **2. Flexibilitatea și adaptabilitatea algoritmilor euristici.** Algoritmii genetici (AG) și tehnicile hibridă (AG +PSO) oferă o flexibilitate și adaptabilitate superioare în abordarea problemelor de optimizare complexe. Acești algoritmi sunt capabili să exploreze un spațiu vast de soluții și să se adapteze rapid la modificările din parametrii problemei. Flexibilitatea acestor algoritmi permite optimizarea eficientă a resurselor și a proceselor, adaptându-se la cerințele dinamice ale pieței. **3. Reducerea ciclurilor de producție și costurilor.** Implementarea metodelor de optimizare, fie ele exacte sau euristice, contribuie semnificativ la reducerea ciclurilor de producție și a costurilor asociate. De exemplu, utilizarea programării cu constrângeri poate optimiza programarea resurselor și secvențele de operații, minimizând astfel timpii de așteptare și maximizând eficiența utilizării echipamentelor. **4. Integrarea tehnologiilor moderne.** Utilizarea tehnologiilor moderne precum IoT și analiza avansată de date în procesul de optimizare permite monitorizarea în timp real și ajustarea proceselor de producție. Aceasta duce la o	probleme cu un număr mare de variabile și constrângeri, necesitând ore sau chiar zile pentru a găsi o soluție optimă. **2. Soluții inconstante în algoritmii euristici** - Algoritmii euristici precum AG și hibrizii (AG +PSO) pot produce soluții inconstante și inexacte, variabile de la o rulare la alta. Aceasta poate duce la incertitudine în implementarea soluțiilor și necesită multiple rulări pentru a obține un set de soluții comparabile. De exemplu, fiecare rulare a unui algoritm genetic poate produce rezultate diferite, ceea ce necesită o evaluare atentă a soluțiilor obținute. **3. Timpul de pregătire a datelor de intrare metodele exacte -** precum CP, necesită un timp considerabil pentru pregătirea datelor de intrare. Aceasta include definirea variabilelor, domeniilor și constrângerilor, care poate fi un proces laborios și consumator de timp. De exemplu, configurarea corectă a unui model CP în CPLEX poate necesita ore de muncă pentru a asigura că toate constrângerile sunt corect definite și integrate. **4. Necesitatea ajustării fine a parametrilor algoritmilor** - Algoritmii euristici și hibrizi necesită ajustări fine ale parametrilor pentru a obține performanțe optime. Aceasta include setarea ratelor de încrucișare și mutație în AG sau a coeficienților cognitivi și sociali în PSO. Ajustarea incorectă a acestor parametri poate duce la performanțe suboptime. De exemplu, o rată de mutație prea mare în AG poate

îmbunătățire semnificativă a calității produselor și a eficienței operaționale. De exemplu, întreținerea predictivă bazată pe IoT poate reduce timpul de inactivitate al echipamentelor și costurile de reparații. **5. Capacitatea de personalizare și scalabilitate.** Algoritmii de optimizare permit personalizarea producției în masă, răspunzând rapid la cerințele specifice ale clienților fără a compromite viteza sau costurile. Aceasta este esențială pentru competitivitatea într-un mediu industrial globalizat. De exemplu, optimizarea prin roiuri de particule (PSO) poate ajusta rapid parametrii de producție pentru a se adapta la cerințele dinamice ale pieței. **6. Aplicabilitate într-o gama largă de domenii.** Metodele și algoritmii de optimizare sunt aplicabili într-o varietate de domenii industriale, de la logistică și lanțuri de aprovizionare până la sănătate și agricultură. Aceasta demonstrează versatilitatea și valoarea lor în îmbunătățirea proceselor și eficienței operaționale în diferite contexte. De exemplu, programarea liniară poate optimiza rutele de transport în logistică, reducând astfel costurile operaționale și îmbunătățind eficiența livrărilor.	transforma algoritmul într-un proces aleatoriu, în timp ce o rată prea mică poate limita capacitatea de explorare a algoritmului. **5. Dependența de capacități computaționale ridicate.** Implementarea metodelor de optimizare avansate necesită capacități computaționale ridicate, inclusiv servere puternice și software specializat. Aceasta poate reprezenta o barieră semnificativă pentru companiile mici și medii care nu dispun de resursele necesare. De exemplu, rularea unui model de optimizare complex în CPLEX poate necesita un hardware specializat pentru a gestiona volumul mare de date și complexitatea calculului. **6. Limitări în scalabilitate pentru metodele exacte -** deși eficiente pentru probleme de dimensiuni moderate, ele pot deveni impracticabile pentru probleme foarte mari sau cu constrângeri foarte complexe. Aceasta poate limita aplicabilitatea lor în contexte industriale cu cerințe dinamice și variabile. De exemplu, programarea liniară poate deveni ineficientă în cazul unor probleme de optimizare la scară largă, unde numărul de variabile și constrângeri crește exponențial.

OPORTUNITĂȚI	AMENINȚĂRI
1. Integrarea cu tehnologii avansate. Integrarea metodelor de optimizare cu tehnologii avansate precum IoT, big data și IA oferă oportunități semnificative pentru îmbunătățirea eficienței operaționale și adaptabilității proceselor de producție. De exemplu, utilizarea senzorilor IoT pentru monitorizarea în timp real a echipamentelor poate permite ajustări dinamice ale proceselor de producție, optimizând astfel eficiența și reducând timpul de inactivitate. **2. Adoptarea metodelor de optimizare hibridă.** Adoptarea metodelor de optimizare hibridă, care combină punctele tari ale diferitelor tehnici, poate conduce la soluții mai robuste și eficiente. De exemplu, combinarea algoritmilor genetici cu optimizarea prin roiuri de particule poate exploata diversitatea genetică și convergența rapidă, oferind soluții de înaltă calitate pentru probleme complexe. **3. Dezvoltarea de algoritmi adaptivi.** Dezvoltarea de algoritmi adaptivi care își ajustează parametrii în funcție de dinamica problemei poate îmbunătăți performanța metodelor de optimizare. De exemplu, algoritmii de învățare automată pot fi utilizați pentru a ajusta dinamica ratelor de crossover și mutație în algoritmii genetici, optimizând astfel procesul de căutare. **4. Personalizarea producției în masă.** Tehnicile avansate de optimizare permit personalizarea producției în masă, răspunzând rapid la cerințele	**1. Complexitatea și dimensiunea problemelor.** Creșterea complexității și dimensiunii problemelor de optimizare poate reprezenta o amenințare semnificativă. Problemele foarte mari sau cu constrângeri foarte complexe pot deveni ineficiente sau impracticabile pentru metodele actuale de optimizare, necesitând dezvoltarea de tehnici și algoritmi noi. De exemplu, problemele de optimizare la scară largă pot depăși capacitatea de calcul a metodelor exacte tradiționale, necesitând soluții alternative. **2. Dependența de resurse computaționale.** Dependența de resurse computaționale ridicate poate limita accesul companiilor mai mici la tehnologiile avansate de optimizare. Aceasta poate crea un dezavantaj competitiv pentru companiile care nu dispun de resursele necesare pentru a implementa și utiliza aceste tehnologii. De exemplu, necesitatea unor servere puternice și software specializat poate reprezenta o barieră semnificativă pentru companiile cu bugete limitate. **3. Incertitudinea rezultatelor în metodele euristice -** obținute prin metodele euristice pot crea dificultăți în implementarea practică a soluțiilor. De exemplu, algoritmii genetici și hibrizi pot produce soluții diferite la fiecare rulare, necesitând multiple rulări și evaluări

specifice ale clienților. Aceasta poate conduce la o satisfacție crescută a clienților și la o loialitate sporită. De exemplu, optimizarea programării și alocării resurselor poate permite ajustări rapide ale liniilor de producție pentru a produce loturi mici de produse personalizate fără a compromite eficiența. **5. Colaborarea și parteneriatele industriale.** Colaborarea și parteneriatele cu alte companii și instituții academice pot stimula inovația și dezvoltarea de soluții de optimizare mai avansate. De exemplu, parteneriatele cu universități pot facilita accesul la cercetări de ultimă oră și la resurse computaționale avansate, îmbunătățind astfel capacitatea de inovare a companiilor. **6. Extinderea aplicabilității în diverse domenii.** Extinderea aplicabilității tehnicilor de optimizare în diverse domenii industriale poate deschide noi oportunități de afaceri și creștere. De exemplu, aplicarea metodelor de optimizare în logistică, industria aeronatutică, industria auto, industria navală, industria electronica, agricultură și sănătate poate conduce la îmbunătățiri semnificative ale eficienței și costurilor operaționale în aceste sectoare. **7. Explorarea spațiului în care algoritmii hibrizi găsesc soluții optime,** prin experimente în ceea ce privesc valorile parametrilor specifici ai acestor algoritmi.	pentru a obține un set de soluții comparabile și fiabile. **4. Schimbările rapide în tehnologie** - pot face ca anumite metode și algoritmi de optimizare să devină rapid depășite. Aceasta poate necesita investiții constante în cercetare și dezvoltare pentru a menține competitivitatea. De exemplu, noile dezvoltări în IA și machine learning pot schimba rapid peisajul tehnologiilor de optimizare, necesitând adaptări rapide și continue. **5. Reglementările și normele industriale stricte** - pot limita flexibilitatea și aplicabilitatea anumitor tehnici de optimizare. Conformitatea cu aceste reglementări poate impune constrângeri suplimentare asupra modului în care sunt dezvoltate și implementate soluțiile de optimizare. De exemplu, reglementările stricte privind siguranța și calitatea în industria auto, aeronautică, si medicală, pot limita utilizarea anumitor algoritmi de optimizare care nu respectă aceste standarde. **6. Costurile asociate cu implementarea și mentenanța sistemelor de optimizare** pot reprezenta o barieră semnificativă pentru adoptarea pe scară largă. Aceste costuri includ nu doar investiția inițială în hardware și software, ci și costurile continue de mentenanță și actualizare. De exemplu, costurile ridicate asociate cu actualizarea continuă a algoritmilor

	și ajustarea parametrilor pot reprezenta o povară financiară pentru multe companii. **7. Dacă nu se găsesc acele moduri în care setările algoritmilor hibrizi să ducă la rezultate bune, constante,** într-un orizont de timp limitat, pe termen lung vor exista erori în procesul de programare a producției, întârzieri în livrarea comenzilor spre clienți, și, implicit, creșterea costurilor unitare a produselor.

(sursa: prelucrare proprie)

CONCLUZII

Într-o lume industrială în continuă schimbare și creștere a competitivității, optimizarea proceselor de producție de serie devine esențială. Inteligența artificială (IA) oferă un set de instrumente și tehnici puternice pentru a aborda provocările complexe și variate ale acestui domeniu. Această lucrare a explorat diverse abordări și algoritmi de optimizare, evidențiind cum pot fi aplicați pentru a îmbunătăți eficiența și productivitatea proceselor industriale.

Programarea producției este una dintre cele mai complexe funcții în gestionarea proceselor de producție. Alegerea tehnologiei și a itinerariului tehnologic, împreună cu definirea procesului de

fabricație, sunt esențiale pentru atingerea obiectivelor de producție. Aceste decizii depind de caracteristicile piesei de prelucrat, cum ar fi forma, dimensiunile, materialul, toleranțele și funcționalitatea. Prin utilizarea IA, aceste variabile pot fi analizate și optimizate în mod eficient, contribuind la îmbunătățirea globală a procesului de producție.

Funcția obiectiv optimizează performanța unui sistem prin minimizarea unei combinații între costul maxim și timpul ponderat, reflectă abordările inovative ale lui René Guinet în domeniul cercetării operaționale. Guinet a jucat un rol esențial în dezvoltarea acestor modele matematice, influențând profund metodele de optimizare și gestionare eficientă a resurselor, contribuind astfel la progresul industriei prin tehnici avansate de planificare și control al producției.

În contextul producției de serie, funcția obiectiv poate include minimizarea costurilor, maximizarea eficienței utilizării resurselor sau reducerea timpului de producție. Alegerea funcției obiectiv și a metodei de optimizare adecvate este crucială pentru obținerea celor mai bune rezultate.

Algoritmii genetici (AG), inspirați de procesele naturale de evoluție și selecție, sunt un exemplu de tehnică de optimizare euristică care poate fi aplicată cu succes în producția de serie. AG începe cu o populație de soluții posibile, fiecare reprezentând o potențială soluție a problemei. Soluțiile sunt evaluate pe baza unei funcții obiectiv, iar cele mai bune soluții sunt selectate pentru a genera noi soluții prin procese de crossover și mutație. Această metodă permite explorarea unui spațiu vast de soluții și găsirea unor configurații eficiente pentru linii de producție.

Optimizarea prin roiuri de particule (PSO) este o altă tehnică euristică eficientă. Inspirat de comportamentul social al roiurilor de păsări sau bancurilor de pești, PSO implică mișcarea particulelor în spațiul soluțiilor, ghidate de propria experiență și de cea a vecinilor. Aceasta permite o convergență rapidă către soluții optime, fiind deosebit de utilă în problemele unde dinamica procesului de producție este complexă și variabilă.

Combinația acestor două metode, algoritmii hibrizi (AG + PSO), aduce beneficii semnificative. Prin combinarea diversității genetice oferite de AG cu convergența rapidă a PSO, algoritmii hibrizi pot îmbunătăți atât viteza, cât și calitatea soluțiilor finale. De exemplu, în optimizarea liniilor de asamblare, AG poate genera diverse configurații inițiale, iar PSO poate rafina aceste configurații pentru a considera factorii dinamici precum cererea de produse sau disponibilitatea echipamentelor. Această abordare combinată permite găsirea rapidă a soluțiilor eficiente și adaptabile.

Metodele exacte, cum ar fi programarea liniară mixtă (MILP) utilizată în CPLEX/OPL, sunt fundamentale pentru problemele bine definite și liniarizabile. MILP implică formularea unei funcții obiectiv și a unui set de constrângeri, rezolvate pentru a găsi valorile optime ale variabilelor de decizie. În producția de serie, MILP este vitală pentru optimizarea resurselor și minimizarea

costurilor, asigurând soluții precise și eficiente. În cazul de față având in vederea complexitatea matricii tridimensionale care a stă la baza implementării funcției obiectiv timpii de rulare au fost prea mari pentru a putea continua rularea algoritmului.

Pe de altă parte, programarea cu constrângeri (CP), o altă metodă exactă, este ideală pentru problemele combinatorii complexe, cum ar fi planificarea și alocarea resurselor. CP folosește un set de variabile, domenii și constrângeri pentru a modela problema, explorând sistematic spațiul soluțiilor posibile. În producția de serie, CP poate optimiza programarea și ordonanțarea sarcinilor, asigurând soluții fezabile care îndeplinesc toate cerințele.

În concluzie, tehnicile și algoritmii de optimizare, fie ei exacți sau euristici, joacă un rol crucial în îmbunătățirea eficienței și productivității proceselor de producție de serie. Metodele exacte oferă soluții precise și fiabile pentru probleme bine definite, în timp ce metodele euristice oferă flexibilitate și adaptabilitate în abordarea problemelor complexe și dinamice. Integrarea acestor tehnici în strategiile de planificare și programare a producției este esențială pentru menținerea competitivității și performanței operaționale în industria modernă.

Analiza detaliată a celor patru metode de rezolvare a problemei de programare a producției a oferit perspective valoroase asupra eficienței și aplicabilității fiecăreia în contextul specific al acestei probleme. Fiecare metodă a fost evaluată în funcție de capacitatea sa de a respecta constrângerile impuse de timp și de a livra o soluție optimă într-un interval de timp rezonabil.

Rezultatele obținute au variat considerabil între metode. CPLEX OPL, utilizând Programarea Liniară Mixtă, nu a reușit să furnizeze o soluție într-un timp rezonabil, sugerând că complexitatea problemei depășește capacitatea acestui model de a găsi o soluție optimă în cadrul unui timp de calcul practic. În contrast, utilizarea Programării cu Constrângeri în CPLEX OPL a produs rezultate mult mai favorabile, respectând termenul limită de șase săptămâni și oferind o soluție optimă în termenii specificați.

Metodele implementate în Matlab au evidențiat și ele diferențe notabile. Algoritmii Genetici, deși au rulat eficient, nu au reușit să respecte constrângerile de timp impuse, indicând necesitatea unei explorări suplimentare a parametrilor algoritmici sau a unei potențiale combinații cu alte metode pentru a îmbunătăți performanța.

În cadrul experimentelor realizate, s-a observat că algoritmii hibrizi(AG+PSO) au demonstrat o capacitate limitată de a respecta condițiile stricte impuse privind intervalul de timp necesar pentru livrarea produselor. Astfel, dintr-un total de șaizeci de rulări efectuate, doar în 3 cazuri algoritmii hibrizi au reușit să genereze soluții viabile care s-au încadrat în limitele temporale stabilite. Aceste

soluții au rezultat în cicluri de producție de 352 ore, 446 de ore și, respectiv, 480 de ore, evidențiind astfel o inconstanță în performanța algoritmului.

Este important de menționat că, deși aceste rezultate sunt în concordanță cu obiectivele stabilite, frecvența scăzută a succesului sugerează necesitatea unei optimizări suplimentare a parametrilor algoritmilor hibrizi. Această constatare subliniază importanța unei abordări continue în cercetare pentru a îmbunătăți consistența și fiabilitatea algoritmilor, astfel încât aceștia să poată furniza în mod constant soluții eficiente în cadrul cerințelor de producție. Rezultatele obținute indică un potențial promițător al metodei hibride, dar și necesitatea unei analize mai aprofundate pentru a înțelege factorii care contribuie la variabilitatea performanței.

Prin urmare, aceste observații subliniază importanța alegerii metodei de optimizare în funcție de caracteristicile specifice ale problemei. De asemenea, ele sugerează că, în situațiile în care constrângerile de timp sunt critice, tehnicile avansate de optimizare cu constrângeri pot oferi soluții mai eficiente decât modelele liniare tradiționale. În plus, rezultatele sugerează necesitatea unei ajustări fine a parametrilor algoritmici și, posibil, a unei reevaluări a modelului de problemă pentru a îmbunătăți șansele de succes în utilizarea metodelor de optimizare.

În concluzie, studiul de față subliniază importanța alegerii metodei de optimizare în funcție de specificul problemei și de constrângerile impuse. Metodele tradiționale, cum ar fi Programarea Liniară, pot deveni ineficiente în fața unor probleme complexe cu multiple constrângeri, în timp ce abordările avansate, cum ar fi Programarea cu Constrângeri oferă alternative viabile, dar care necesită o setare atentă a parametrilor pentru a obține rezultate optime.

Abordările hibride, cum ar fi combinația dintre Algoritmii Genetici și PSO, prezintă un potențial semnificativ în optimizarea problemelor complexe, oferind un echilibru între explorare și exploatare. Cu toate acestea, succesul acestor metode este limitat și inconstant și depinde în mare măsură de modul în care sunt integrate și de capacitatea lor de a se adapta la specificul problemei.

CONTRIBUȚII PROPRII

Una dintre cele mai importante contribuții ale tezei constă în integrarea și analiza comparativă a metodologiilor exacte și euristice în programarea producției. Studiul aduce o perspectivă valoroasă asupra modului în care aceste două abordări, fiecare cu avantajele și limitările proprii, pot fi utilizate pentru optimizarea producției în diverse contexte industriale. Prin utilizarea algoritmilor genetici și hibrizi, s-a demonstrat eficiența acestor metode în reducerea timpilor de procesare și optimizarea resurselor, oferind în același timp un cadru flexibil, adaptabil la nevoile specifice ale fiecărei industrii.

O altă contribuție deosebită a tezei este dezvoltarea unui cadru analitic detaliat pentru aplicarea modelului Guinet. S-a demonstrat cum acest model poate aduce îmbunătățiri semnificative în reducerea ciclurilor de producție, oferind soluții clare și aplicabile pentru companiile care urmăresc optimizarea proceselor prin intermediul tehnologiilor avansate, cum ar fi algoritmii de IA. Această contribuție susține nu doar eficiența proceselor de producție, ci și adaptabilitatea lor la diverse industrii.

O a treia contribuție deosebită o reprezintă găsirea unei metode de rezolvare a modelului Guinet , prin intermediul CPLEX OPL Constrain Programming, care a furnizat minimizarea valorii funcției obiectiv, rezolvarea problemei ordonanțării comenzilor în detaliu, reușșind cel mai mic timp de rulare.

A. Contributii teoretice

1. Integrarea metodologiilor exacte și euristice. Teza explorează modul în care metodologiile exacte și euristice pot fi integrate pentru optimizarea producției. Printr-o analiză comparativă, se demonstrează avantajele fiecărei abordări, precum reducerea timpilor de procesare și optimizarea resurselor. Algoritmii genetici și hibrizi se dovedesc eficienți în acest context, oferind un cadru flexibil și adaptabil pentru diverse industrii. Această integrare permite companiilor să utilizeze metode personalizate pentru nevoile lor specifice, asigurând o producție mai eficientă și competitivă în mediul industrial actual.

2. Dezvoltarea unui cadru analitic pentru modelul Guinet. Teza oferă un cadru analitic detaliat pentru aplicarea modelului Guinet în optimizarea producției. S-a demonstrat că modelul aduce îmbunătățiri semnificative în reducerea ciclurilor de producție, oferind soluții clare pentru companiile care doresc să îmbunătățească eficiența proceselor lor. Prin utilizarea tehnologiilor avansate, precum algoritmii de inteligență artificială, acest model susține nu doar eficiența producției, ci și adaptabilitatea sa la diverse industrii, contribuind astfel la dezvoltarea unui cadru operațional modern și competitiv.

3. Compararea metodelor exacte și euristice. Teza prezintă o analiză comparativă între metodele exacte și cele euristice, evidențiind avantajele și dezavantajele fiecăreia în optimizarea producției. Algoritmi precum CPLEX/OPL Constrain Programming și Algoritmul Genetic Hybrid (AG + PSO) sunt utilizați pentru a demonstra flexibilitatea și eficiența acestor metode în diverse contexte industriale. Această comparație subliniază importanța alegerii metodologiei adecvate pentru a obține rezultate optime în programarea producției și pentru a crește eficiența operațională.

4. Abordarea integrată a cercetării teoretice și aplicate. Lucrarea se distinge prin abordarea integrată a cercetării teoretice și aplicate, contribuind la avansul cunoștințelor în domeniul optimizării producției. Validarea utilității algoritmilor de IA în sectorul industrial se bazează pe un echilibru între teorie și practică. Această abordare holistică oferă un cadru solid pentru cercetători și practicieni în înțelegerea și aplicarea IA în producție.

5. Analiza tendințelor și provocărilor în adoptarea IA. Teza analizează principalele tendințe și provocări cu care se confruntă companiile în adoptarea tehnologiilor IA. Această analiză oferă un ghid valoros pentru manageri și decidenți, facilitând înțelegerea complexității implementării IA în procesele operaționale și strategice. Prin abordarea acestor provocări, companiile pot rămâne competitive pe piața globală, înțelegând mai bine cum să îmbine resursele umane și tehnologice pentru a maximiza eficiența și a îmbunătăți procesele de producție.

6. Modelarea programării producției de serie. Capitolul 3 reprezintă contribuția centrală a tezei, concentrându-se pe modelarea programării producției de serie prin algoritmi de inteligență artificială. Printr-o analiză detaliată, se oferă soluții practice pentru optimizarea producției de serie. Se abordează aspecte precum reducerea timpilor de așteptare și îmbunătățirea fluxurilor de producție, demonstrând modul în care IA poate contribui la eficientizarea operațiunilor în mediul industrial, oferind o bază solidă pentru implementarea acestor tehnologii în companii.

7. Utilizarea algoritmilor genetici și hibrizi. Lucrarea explorează utilizarea algoritmilor genetici și hibrizi pentru optimizarea producției. Acești algoritmi s-au dovedit a fi eficienți în reducerea timpilor de procesare și optimizarea resurselor, oferind un cadru flexibil care se poate adapta la nevoile specifice ale fiecărei industrii. Aceasta permite companiilor să-și optimizeze procesele, să reducă costurile și să îmbunătățească eficiența operațională, contribuind astfel la creșterea competitivității în mediul industrial actual.Metodologia de cercetare aplicată: Prezentarea unei metodologii riguroase pentru analiza aplicării IA în producție, bazată pe interviuri ghidate cu experți din industrie.

8. Adaptabilitatea la diverse industrii. Teza subliniază adaptabilitatea algoritmilor de IA și a modelului Guinet la diverse industrii. Prin oferirea unui cadru flexibil și personalizabil, companiile pot implementa aceste tehnologii în funcție de nevoile lor specifice, asigurând o optimizare eficientă a proceselor de producție. Aceasta permite companiilor să se adapteze mai rapid la schimbările din mediul de afaceri și să rămână competitive pe o piață dinamică.

9. Metodologia de cercetare aplicată. Teza prezintă o metodologie riguroasă pentru analizarea aplicării IA în programarea producției de serie. Prin interviuri ghidate cu experți din industrie, s-a obținut o înțelegere profundă a modului în care aceste tehnologii pot fi integrate în industriile locale.

Această abordare metodologică a permis evidențierea legăturii dintre cercetarea teoretică și aplicațiile practice, oferind o bază solidă pentru implementarea IA în procesele de producție.

10. Ghid pentru manageri și decidenți. Teza oferă un ghid valoros pentru manageri și decidenți în procesul de implementare a tehnologiilor IA în producție. Prin evidențierea provocărilor, beneficiilor și a soluțiilor practice, lucrarea facilitează înțelegerea complexității și a oportunităților asociate cu utilizarea IA în procesele operaționale. Aceasta ajută companiile să ia decizii informate pentru a rămâne competitive și pentru a-și îmbunătăți eficiența și flexibilitatea operațională.

B. Contributii practice

1. Rezolvarea modelului Guinet cu CPLEX OPL. Teza prezintă o metodă de rezolvare a modelului Guinet folosind CPLEX OPL Constrain Programming. Această abordare a permis minimizarea valorii funcției obiectiv și optimizarea ordonării comenzilor, obținând cel mai scurt timp de rulare. Aceasta este o contribuție majoră în domeniul producției, demonstrând cum instrumentele avansate de programare pot rezolva probleme complexe, permițând astfel companiilor să optimizeze procesul de producție, să reducă costurile și să îmbunătățească eficiența operațională.

2. Validarea algoritmilor IA în producția industrială. Studiul de caz asupra companiei ABC validează eficiența algoritmilor de inteligență artificială în producția industrială românească. Această validare merge dincolo de teorie, demonstrând aplicarea practică a acestor tehnologii și evidențiind beneficiile lor în contextul actual al industriei. Prin interviuri ghidate și aplicații practice, se oferă o perspectivă relevantă pentru companiile care intenționează să implementeze tehnologiile IA, contribuind astfel la creșterea competitivității și la îmbunătățirea proceselor operaționale.

3. Soluții practice pentru implementarea IA. Pe lângă aportul academic, teza oferă soluții practice pentru implementarea IA în managementul producției. Aceste soluții sunt aplicabile și adaptabile pentru diverse tipuri de companii, permițând optimizarea proceselor, reducerea costurilor și îmbunătățirea flexibilității operaționale. Astfel, lucrarea devine un ghid util atât pentru cercetători, cât și pentru practicieni, arătând cum tehnologiile IA pot fi integrate eficient în procesele de producție pentru a obține avantaje competitive semnificative.

4. Studiu de caz: compania ABC. Studiul de caz al companiei ABC oferă un exemplu practic al implementării metodelor exacte în producția de serie. Prin interviuri ghidate și analizarea datelor de producție, cercetarea demonstrează aplicabilitatea algoritmilor de IA în procesele de producție. Aceasta evidențiază modul în care aceste tehnologii pot fi adaptate și aplicate în contextul specific al unei companii, oferind perspective valoroase pentru alte organizații care doresc să își îmbunătățească eficiența producției.

5. Integrarea tehnologiilor avansate. Teza analizează integrarea tehnologiilor avansate în procesele operaționale, subliniind atât avantajele, cât și provocările întâlnite în adoptarea acestor soluții inovatoare. Această analiză oferă un ghid valoros pentru manageri, ajutându-i să înțeleagă riscurile și oportunitățile asociate cu utilizarea IA în producție. Prin adoptarea acestor tehnologii, companiile pot îmbunătăți eficiența și flexibilitatea operațională, devenind astfel mai competitive pe piața globală.

6. Reducerea timpilor de așteptare. Prin aplicarea modelului Guinet, teza demonstrează cum companiile pot reduce timpii de așteptare în producție. Modelul oferă soluții avansate pentru optimizarea ciclurilor de producție, permițând companiilor să-și îmbunătățească fluxurile de producție și să utilizeze mai eficient resursele. Aceasta contribuie la creșterea eficienței operaționale, reducerea costurilor și îmbunătățirea competitivității pe piața globală.

7. Reducerea costurilor prin IA. Teza demonstrează cum algoritmii de IA pot contribui la reducerea costurilor în producție. Prin optimizarea proceselor și îmbunătățirea utilizării resurselor, companiile pot reduce cheltuielile operaționale și pot crește profitabilitatea. Această abordare oferă un avantaj competitiv semnificativ, permițând companiilor să își adapteze strategiile de producție pentru a răspunde mai bine cerințelor pieței și pentru a îmbunătăți eficiența operațională.

8. Minimizarea valorii funcției obiectiv. Prin utilizarea CPLEX OPL Constrain Programming, teza reușește să minimizeze valoarea funcției obiectiv în contextul ordonanțării comenzilor. Această minimizare este esențială pentru îmbunătățirea eficienței proceselor de producție, permițând companiilor să își optimizeze programarea producției, să reducă timpii de așteptare și să îmbunătățească fluxurile de lucru. Acest lucru conduce la o producție mai eficientă și la o mai bună utilizare a resurselor disponibile.

9. Flexibilizarea operațională. Prin implementarea tehnologiilor IA, companiile pot crește flexibilitatea operațională. Aceasta permite ajustarea rapidă a proceselor de producție în funcție de cerințele pieței și de schimbările în mediul de afaceri. Lucrarea demonstrează cum companiile pot utiliza aceste tehnologii pentru a îmbunătăți capacitatea de reacție la cerințele clienților, optimizând astfel producția și maximizând eficiența**.**

10. Eficientizarea proceselor de producție. Lucrarea contribuie la creșterea eficienței proceselor de producție prin rezolvarea în CPLEX a modelului Guinet. Prin reducerea timpilor de așteptare, optimizarea fluxurilor de producție și utilizarea eficientă a resurselor, companiile pot îmbunătăți semnificativ eficiența operațională. Aceasta are un impact direct asupra competitivității pe piața globală, permițând companiilor să livreze produse mai rapid și mai eficient.

Așadar, prin studiul de caz asupra companiei ABC, teza validează eficiența modelului Guinet adaptat în producția industrială românească. Această validare nu se limitează la o abordare teoretică, ci este susținută de aplicări practice și interviuri ghidate, oferind o perspectivă relevantă pentru companiile care doresc să implementeze tehnologii de ultimă generație.

Cercetarea evidențiază principalele tendințe și provocări cu care se confruntă companiile în adoptarea tehnologiilor IA. Această analiză oferă un ghid valoros pentru manageri și decidenți, facilitând înțelegerea complexității implementării IA în cadrul proceselor operaționale și strategice, cu scopul de a rămâne competitivi pe piața globală.

Pe lângă îmbogățirea literaturii de specialitate, teza oferă soluții practice pentru implementarea IA în managementul producției. Aceste soluții sunt aplicabile și adaptabile, permițând companiilor să optimizeze procesele și să îmbunătățească eficiența și flexibilitatea operațională. Astfel, teza devine un ghid util atât pentru cercetători, cât și pentru practicieni.

Direcții viitoare de cercetare

În contextul actual, unde cerințele pieței sunt în continuă schimbare, iar eficiența și optimizarea proceselor de producție devin priorități critice, se impune o aprofundare a cercetării în domeniul optimizării parametrilor algoritmilor de programare a producției. Recomandăm continuarea investigațiilor în direcția ajustării și optimizării parametrilor care guvernează comportamentul algoritmilor Genetici (AG) și a metodelor hibride avansate, cum ar fi combinația dintre Algoritmii Genetici și Optimizarea prin Roiuri de Particule (PSO).

În privința optimizării parametrilor în Algoritmii Genetici (AG):

Unul dintre aspectele esențiale care influențează performanța Algoritmilor Genetici este alegerea corectă a parametrilor de bază, cum ar fi mărimea populației, rata de încrucișare și rata de mutație.

- **Mărimea populației**: Acest parametru determină numărul total de soluții potențiale (cromozomi) evaluate la fiecare iterație. O populație mai mare poate duce la o explorare mai eficientă a spațiului de căutare, reducând riscul de a se bloca în optime locale. Cu toate acestea, o populație mai mare necesită și resurse computaționale suplimentare și poate duce la o creștere a timpului de execuție al algoritmului. Cercetările viitoare ar trebui să se concentreze pe identificarea unei mărimi optime a populației care să maximizeze performanța algoritmului fără a crește excesiv cerințele de calcul.
- **Rata de încrucișare:** Rata de încrucișare, care controlează frecvența cu care sunt combinate perechi de cromozomi pentru a genera noi soluții, joacă un rol crucial în asigurarea diversității

populației. O rată de încrucișare prea mare poate perturba structurile bune de soluții, în timp ce o rată prea mică poate duce la convergență prematură. Direcțiile viitoare de cercetare ar trebui să exploreze modul în care variația dinamică a ratei de încrucișare în funcție de stadiul algoritmului ar putea îmbunătăți eficiența și eficacitatea acestuia.

- **Rata de mutație:** Mutația introduce variații suplimentare în populație, permițând algoritmului să exploreze noi zone ale spațiului de soluții. În mod similar cu rata de încrucișare, rata de mutație trebuie ajustată cu atenție. O rată de mutație prea mare poate transforma algoritmul într-un proces aleatoriu, în timp ce o rată prea mică poate limita capacitatea algoritmului de a evita optimele locale. Cercetarea ar trebui să se concentreze pe dezvoltarea unor scheme adaptative de mutație, care să ajusteze rata în funcție de dinamica procesului evolutiv.

În privința optimizării parametrilor în algoritmii hibrizi (AG + PSO):

Pentru algoritmii hibrizi, care combină Algoritmii Genetici cu Optimizarea prin Roiuri de Particule, există alți parametri cheie care necesită ajustare și optimizare.

- **Mărimea roiului:** La fel ca în cazul mărimii populației din AG, mărimea roiului în PSO influențează echilibrul între explorare și exploatare. Un roi mai mare poate oferi o explorare mai completă a spațiului de soluții, dar crește și complexitatea calculului. Cercetările viitoare ar trebui să analizeze impactul mărimii roiului asupra performanței generale a algoritmului hibrid și să identifice dimensiunile optime pentru diverse tipuri de probleme de producție.
- **Factorul inerției (w):** Factorul de inerție controlează influența vitezei anterioare a particulelor asupra vitezei actuale. Un factor de inerție mare favorizează explorarea, în timp ce un factor mic favorizează exploatarea. Studii viitoare ar trebui să investigheze utilizarea unui factor de inerție variabil, care să scadă treptat pe măsură ce algoritmul se apropie de convergență, pentru a asigura un echilibru optim între explorare și exploatare.
- **Coeficienții cognitivi și sociali (c1 și c2):** Acești coeficienți determină cât de mult sunt particulele atrase de pozițiile lor anterioare de succes și de poziția globală de succes a roiului. Cercetarea ar trebui să se concentreze pe optimizarea acestor coeficienți pentru a maximiza performanța algoritmului hibrid. De exemplu, se poate investiga impactul ajustării dinamice a coeficienților cognitivi și sociali în funcție de evoluția procesului de căutare.
- **Viteza particulelor:** Controlul vitezei este esențial pentru a evita ca particulele să depășească regiuni de interes din spațiul de căutare. Limitarea vitezei particulelor, cunoscută sub numele de velocity clamping, poate îmbunătăți stabilitatea și eficiența algoritmului. Cercetările

viitoare ar trebui să exploreze impactul diferitelor strategii de velocity clamping asupra performanței algoritmului hibrid.

Influența constrângerilor asupra performanței algoritmilor. Un alt aspect critic care necesită o atenție deosebită este modul în care constrângerile impuse problemelor de producție influențează performanța algoritmilor. Este esențial să se realizeze o analiză detaliată a modului în care diverse tipuri de constrângeri (de timp, de resurse, de calitate etc.) afectează capacitatea algoritmilor de a găsi soluții optime. Acest lucru ar putea implica dezvoltarea unor modele de optimizare mai robuste, capabile să se adapteze la diverse scenarii de producție.

De asemenea, ar putea fi benefică explorarea altor tehnici de optimizare avansate, cum ar fi algoritmii evolutivi sau alte metode metaeuristice. Aceste metode au demonstrat rezultate promițătoare în alte domenii și ar putea oferi soluții inovatoare și eficiente pentru problemele complexe de programare a producției.

Importanța unui proces iterativ de testare și ajustare. În final, este esențial ca orice metodă de optimizare utilizată să fie susținută de un proces iterativ de testare și ajustare. Acest proces permite rafinarea continuă a modelului și îmbunătățirea performanței în raport cu obiectivele stabilite. O abordare iterativă asigură că algoritmii pot fi ajustați în funcție de rezultatele obținute, permițând astfel dezvoltarea unor soluții optimizate, capabile să răspundă eficient cerințelor pieței și să asigure succesul pe termen lung al organizației. În acest sens, colaborarea strânsă între cercetători și practicieni va juca un rol esențial în realizarea unor progrese semnificative în domeniul optimizării programării producției.

Limitele cercetării. În cadrul acestei cercetări, am identificat o serie de limitări care au influențat atât procesul de obținere a rezultatelor, cât și calitatea acestora. Aceste limitări sunt esențiale pentru o înțelegere completă a contextului în care au fost realizate experimentele și pentru o interpretare corectă a rezultatelor obținute. De asemenea, ele oferă o direcție clară pentru cercetările viitoare, care ar trebui să abordeze aceste provocări pentru a îmbunătăți performanța algoritmilor și relevanța soluțiilor propuse:

a) Mărimea matricei tridimensionale. Prima limitare majoră a cercetării a fost impusă de mărimea considerabilă a matricei tridimensionale utilizate în implementarea funcției obiectiv. Această matrice este esențială pentru modelarea complexă a spațiului de căutare și pentru asigurarea că toate constrângerile sunt luate în considerare în procesul de optimizare. Cu toate acestea, dimensiunea sa mare a condus la un timp de rulare excesiv de lung pentru găsirea unei soluții viabile prin CPLE OPL Mixt Integer Mixt Integer Linear Programming.

În mod specific, complexitatea calculului asociat cu această matrice a necesitat resurse computaționale semnificative, ceea ce a făcut ca timpul de execuție să devină nefezabil în contextul aplicabilității practice. Această situație este problematică, mai ales într-un mediu industrial unde rapiditatea și eficiența sunt critice. Astfel, această limitare a subliniat necesitatea unor metode mai eficiente de reducere a dimensiunii problemei, fie prin simplificarea modelului matematic, fie prin implementarea unor tehnici avansate de calcul paralel sau distribuirea sarcinilor pe mai multe resurse computaționale. Abordarea acestor aspecte ar putea conduce la o îmbunătățire semnificativă a timpului de răspuns și, implicit, la o mai bună aplicabilitate a soluției în contexte reale.

b) Variația rezultatelor în cazul Algoritmilor Genetici (AG). A doua limitare semnificativă a fost observată în cazul utilizării AG. Deși acești algoritmi sunt cunoscuți pentru capacitatea lor de a explora eficient spații de căutare mari și complexe, rezultatele obținute au prezentat o variație considerabilă de la o rulare la alta. Această variație este problematică, deoarece reduce predictibilitatea și fiabilitatea soluțiilor obținute, făcând dificilă încadrarea în intervalul de timp stabilit pentru livrarea produselor.

Mai mult, AG nu au reușit să respecte în mod consistent condițiile de timp impuse, ceea ce pune sub semnul întrebării acuratețea și relevanța datelor de ieșire. Acest aspect indică o posibilă nevoie de ajustare a parametrilor algoritmici, cum ar fi mărimea populației, rata de mutație sau rata de încrucișare, pentru a îmbunătăți performanța algoritmului. De asemenea, ar putea fi necesară integrarea unor metode suplimentare de optimizare sau hibridizarea AG cu alte tehnici pentru a stabiliza rezultatele și a asigura conformitatea cu cerințele de timp. Această limitare evidențiază, de asemenea, importanța unei calibrări adecvate a algoritmului în funcție de specificul problemei de optimizare abordate.

c) Inconsistența rezultatelor în cazul algoritmilor hibrizi

Deși utilizarea algoritmilor hibrizi, care combină Algoritmii Genetici cu Optimizarea prin Roiuri de Particule (PSO), a arătat un potențial promițător în abordarea problemelor complexe, rezultatele au fost, din păcate, inconstante. În cadrul experimentelor, doar în trei cazuri din șaizeci, algoritmii hibrizi au reușit să furnizeze soluții care respectă condiția de încadrare în intervalul de timp pentru livrarea produselor. Acest fapt subliniază un alt nivel de variabilitate care ridică întrebări cu privire la robustețea și fiabilitatea acestor metode.

Inconsistența rezultatelor în funcție de rulare sugerează că algoritmii hibrizi sunt sensibili la setările inițiale ale parametrilor, cum ar fi mărimea roiului, factorul de inerție și coeficienții cognitivi și sociali. Aceasta implică necesitatea unei cercetări mai aprofundate pentru a înțelege mai bine dinamica acestor algoritmi și pentru a dezvolta strategii de ajustare automată a parametrilor în timpul

rulării, în funcție de performanțele observate. În plus, acuratețea datelor de ieșire obținute de algoritmii hibrizi a fost considerată nesatisfăcătoare în anumite cazuri, ceea ce indică nevoia de rafinare a metodei de evaluare a soluțiilor generate.

Această limitare subliniază, de asemenea, necesitatea unor metode mai avansate de testare și validare a algoritmilor hibrizi, pentru a se asigura că rezultatele sunt consistente și fiabile, independent de variațiile minore ale condițiilor inițiale. Posibile soluții ar putea include implementarea unor mecanisme de feedback care să permită algoritmului să învețe din performanțele sale anterioare și să ajusteze comportamentul în consecință.

În concluzie, limitările identificate în cadrul acestei cercetări oferă un punct de plecare important pentru direcții viitoare de cercetare. Necesitatea de a optimiza performanța algoritmilor prin ajustarea parametrilor și prin implementarea unor metode mai robuste și eficiente este evidentă. Abordarea acestor provocări va contribui nu doar la îmbunătățirea algoritmilor existenți, ci și la dezvoltarea unor noi tehnici de optimizare capabile să răspundă cerințelor complexe și variate ale producției moderne.

BIBLIOGRAFIE TEZA

[1] F. Toader, „Production Scheduling in Flexible Manufacturing Systems: A State of the Art Survey,” *Journal of Electrical Engineering, Electronics, Control and Computer Science ,* vol. 3, nr. 1, pp. 1-6, 2017.

[2] S. Gao și P. Sui, „Production Management,” în *Lean Construction Management*, New York, Springer, 2014, pp. 11-25.

[3] E. Rater și S. Nader, „The Use of Lean Management Tools in Production Companies with Implemented Total Quality Management,” *European Research Studies Journal,* vol. XXV, nr. 3, pp. 357-368, 2022.

[4] S. Adeniran și M. Omolayo, „The Role of Production Planning in Enhancing an Efficient Manufacturing System - An Overview,” în *E3S Web of Conferences* , 2021.

[5] A. Kosieradzka , U. Kakol și A. Krupa, „The Development of Production Management Concepts,” *Foundations of Management ,* vol. 3, nr. 2, pp. 1-6, 2011.

[6] J. Hinckeldeyn, R. Dekkers și J. Kreutzfeldt, „Application of Production Management Principles to Engineering Process,” în *Industrial Engineering an Engineering Management* , IEEE International Conference Online, 2010.

[7] M. Sreekumar, M. Chhabra și S. Yadav, „Production and Operations Management: Challenges and Trents beyond 2020,” în *Paradigm Shift in Business, Economy and Society in*

New Millenium , Udaipur, Pacific Academy of Higher Education and Research University, 2020, pp. 257-274.

[8] C. Chen, „Identifying the promising production planning and scheduling method for manufacturing in Industry 4.0. - a literature review," *Taylor & Francis, Production& Manufacturing Research: An Open Access Journall,* vol. 11, nr. 1, 2023.

[9] K. Stecke și R. Parker, „Flexible Automation," în *Innovations in Competitive Manufacturing* , London, Kluver Academic Publishers , 2000, pp. 109-116.

[10] M. Pristavka, M. Kotorova și R. Savov, „Quality Contronl in Production Processes," *Acta Technologica Agriculture,* vol. 19, nr. 3, pp. 77-83, 2016.

[11] P. Zwiech , „Sustainable production," în *Organizing Sustainable Development*, New York, Routledge, 2023, pp. 120-132.

[12] M. Carvalho, „Production Scheduling on Practical Problems," în *Production Scheduling*, Rjeka, Intech Open, 2012.

[13] A. Semenova și E. Ozdamirova , „The Future of Work: Automation, AI and Information Tehnology," în *Web of Conferences*, 2023.

[14] P. Kess și V. Isoherranen, „Production planning optimization and challenges in steel production," *International Journal of Modelling in Operations Management,* vol. 6, nr. 1/2, 2016.

[15] D. Romero și G. Von Cieminski, „Advances in Production Management System : Issues, Trends, and Vision Towards 2030," în *Advancing Research in Information and Communication Technology*, New York, Springer, 2021, pp. 194-221.

[16] A. Akdogan și A. S. Vanli, „Mass Production and Industry 4.0," în *Mass Production Processes* , London, Intech Open, 2020.

[17] A. Almeida și C. Ramos, „Simultaneous manufacturing in batch proction," în *Assembly and Task Planning, Proceedings of the IEEE International Symposium on* , Besancon, 2003.

[18] A. Saniuk și R. Waszkowski, „Make-to-order manufacturing- new approach to management of manufacturing processes," în *Series Materials Science and Engineering* , 2016.

[19] J. Langstrand , „Pettersen, J. : Defining lean production : some conceptual and practical issues," *TQM Journal,* vol. 21, nr. 2, pp. 127-142, 2009.

[20] F. Chen și A. Everett, „The impact of flexible manufacturing systems on productivity and quality," *IEEE Transactions on Engineering Management ,* vol. 38, nr. 1, pp. 33-45, 1991.

[21] M. Sreekumar și M. Chhabra, „Production and Operations Management: Challenges and Trends beyond 2020," în *Paradigm Shift in Business, Economy and Society in New*

Millennium, Udaipur, Pacific Academy of Higher Education and Research University, 2020, pp. 257-274.

[22] H. Pourasiabi, „Just in Time (JIT) Production and Supply Chain Management," în *International Iron and Steel Symposium*, Karabuk, 2012.

[23] Y. Zhang , W. Wang și N. Wu , „IoT Enabled Real Time Production Performance Analysis Exception Diagnosis Model," *IEEE Transactions on Automation Science and Engineering ,* vol. 13, nr. 3, pp. 1-15, 2015.

[24] W. Ding, „Study of Smart Warehouse Management System Based on IoT," în *Intelligence Computation and Evolutionary Computation* , New York, Springer, 2013, pp. 203-207.

[25] R. Falkenberg, M. Masoudinejad, M. Buschhoff, A. Venkatapathy, D. Friese și M. ten Hompe, „PhyNetLab: An IoT-Based Warehouse Testbed," în *Proceedings of the Federated Conference on Computer Science and Information Systems* , Dortmund, 2017.

[26] A. Boza, C. Beatriz, L. Cuenca și F. Alarcon, „Internet of Things Applications in Production Systems," în *Proceedings of the 17th International Conference on Enterprise Information Systems - Volume 2: ICEIS*, Barcelona, 2015.

[27] E. Mardiani, D. Riswandi, D. Suprayitno și K. Khamaludin, „Implementation of Internet of Things in the Production Process of MSMEs: Quality Improvement and Process Control," *Jurnal Informasi & Teknologi (JidT),* vol. 6, nr. 1, pp. 310-316, 2024.

[28] T. Kalsoom, S. Ahmed, P. Rafi-Ui-Shan și M. Azmat, „Impact of IoT on Manufacturing Industry 4.0: A New Triangular Systematic Review," *MDPI,* vol. 13, nr. 22, 2021.

[29] J. Alzahrani, „Job-Shop Scheduling Optimization with Stochasting Processing Times," *International Journal of Engineering Technologies and Management Research ,* vol. 6, nr. 1, pp. 73-83, 2020.

[30] W. Abdullah, „Solving Job-Shop Scheduling Problem Using Genetic Algorithm Approach," *Journal of the college of basic education ,* vol. 17, nr. 70, pp. 241-253, 2022.

[31] C. Ceren, A. Enes și O. Sahingoz, „Job Shop Scheduling Problem and Solution Algorithms: A Review," în *2020 11th International Conference on Computing, Communication and Networking Technologies (ICCCNT)*, Kharagpur, 2020.

[32] S. Dao și R. Marian, „Modeling and Optimization of Precedence-Constrained Production Sequencing and Scheduling Using Multi-Objective Genetic Algorithms," în *Conference: The World Congress on Engineering*, London, 2011.

[33] E. Alpaydin, Introduction to Machine Learning, Massachusetts: The MIT Press, 2020.

[34] R. Cupek, A. Ziebinski, L. Huczala și H. Erdongan, „Agent-based manufacturing execution systems for short-series production scheduling," *Computers in Industry,* vol. 82, 2016.

[35] L. Morar, E. Westkamper și I. Abrudan, Planning and Operation of Production Systems, Stuttgart : Fraunhofer IRB Verlag, 2007.

[36] K. Metaxiotis, D. Askounis și J. Psarras, „Expert systems in production planning and scheduling: A state-of-the-art survey," *Journal of Intelligent Manufacturing ,* vol. 13, pp. 253-260, 2002.

[37] G. Panaitescu, Modelarea și simularea sistemelor de producție- Curs pentru ID, Ploiești: Universitatea de Petrol și Gaze, 2007.

[38] M. Zhang și Z. Zhou, „A review on multi-label learning algorithms," *IEEE transactions on knowledge and data engineering ,* vol. 26, nr. 8, pp. 1819-1837, 2018.

[39] I. Abrudan și D. Cândea, Manual de inginerie economică. Ingineria și managementul sistemelor de producție, Cluj-Napoca: Dacia, 2002.

[40] Y. Zhang, W. Wenbo, N. Wu și C. Qian, „IoT-Enabled Real-Time Production Performance Analysis and Exception Diagnosis Model," *IEEE Transactions on Automation Science and Engineering,* vol. 13, nr. 3, pp. 1-15, 2015.

[41] W. Wang, H. Yang, Y. Zhang și J. Xu, „IoT-enabled real-time energy efficiency optimisation method for energy-intensive manufacturing enterprises," *International Journal of Computer Integrated Manufactoring ,* vol. 31, nr. 4-5, pp. 362-379, 2018.

[42] Y. Zhang, S. Liu, H. Yang și D. Huisingh, „The Internet of Things enabled real-time," *Journal of Cleaner Production,* vol. 185, nr. 1, pp. 562-575, 2018.

[43] S. Duy Dao, K. Abhary și R. Marian, „Optimisation of assembly scheduling in VCIM systems using genetic algoritm," *Journal of Industrial Engineering International ,* vol. 13, pp. 275-296, 2017.

[44] S. Duy Dao, K. Abhary și R. Marian, „An innovative framework for designing genetic algorithm structures," *Elsevier - Expert Systems with Applications,* vol. 90, pp. 196-208, 2017.

[45] S. Duy Dao, Modelling and Intelligent Optimisation of Production Scheduling in VCIM Systems, New York: Springer , 2018.

[46] S. Duy Dao, R. Marian și L. Luong , „Hybrid Genetic Algorithm Optimisation of Distribution Networks—A Comparative Study," în *Intelligent Control and Innovative Computing*, New York, Springer, 2012, pp. 109-122.

[47] A. Guinet, M. Solomon , P. Keida și A. Dussauchoy, „A computational study of heuristics for two-stage," *International Journal of Production Research,* vol. 34, nr. 5, pp. 1399-1415, 2010.

[48] S. Othman, H. Zgaya, M. Dotoly și S. Hammadi, „An agent-based Decision Support System for resources scheduling in Emergency Supply Chains,” *Control Engineering Practice,* nr. 59, pp. 27-43, 2017.

[49] P. Tavinder și S. Simranjit, „Review of supply chain management in manufacturing organizations,” *JAC: A Journal of Composition Theory,* vol. XII, nr. VII, pp. 1000-1007, 2024.

[50] N. Pretty, „New opportunities for the redesign of agricultural and food systems,” *Agriculture and Human Values,* vol. 37, pp. 629-630, 2020.

[51] A. Guinet, R. Bachouch și S. Hajri-Gabouj, „An integerlinear model for hospital bed planning,” *Production Economics,* nr. 140, pp. 833-843, 2012.

[52] A. Guinet și R. Faccincani, „Hospital's vulnerability assessment,” în *International Conference on Industrial Engineering and Systems Management (IESM)*, 2015 .

[53] Y. Yoldas, A. Onen și S. Muyeen, „Enhancing smart grid with microgrids: Challenges and opportunities,” *Renewable and Sustainable Energy Reviews,* vol. 72, pp. 205-214, 2017.

[54] A. Guinet, „Transports industriels routiers, un problème d’affectation avec ŕemploi sous contraintes,” *RAIRO - Operations Research,* vol. 18, nr. 4, 1984.

[55] N. Mărginean , S. Vultur și D. Marincaș, Introducere în Inteligența Artificială, Cluj-Napoca: Risoprint, 2007.

[56] S. Dreiseitl și L. Ohno-Machado, „Logistic regression and artificial neural network classification models,” *Journal of biomedical informatics,* vol. 35, pp. 5-6, 2002.

[57] Russel S. și P. Norvig, Artificial Intelligence - A modern Approach, London: Pearson Prentice Hall, 2020.

[58] P. Winston, Artificial Inteligence, Boston: Addison-Wesley, 1992.

[59] L. Asadzadeh, „A local search genetic algorithm for the job shop scheduling problem with intelligent agents,” *Computers & Industrial Engineering ,* vol. 85, 2015.

[60] M. Gen și R. Gheng, Genetic Algorithms and engeneering optimization, Boston: Wiley Series in Engineering Design and Automation, 2000.

[61] F. Chircu, Teză de doctorat: Contribuții privind dezvoltarea unui sistem inteligent de planificare automată a producției în linii flexibile de fabricație, Ploiești: Universitatea de Petrol și Gaze Ploiești , 2017.

[62] P. Kees și V. Isoherranen, „Production Planning optimisation and challenges in steel production,” *International Journal of Modelling in Operations Management,* vol. 6, nr. 19, pp. 1-2, 2016.

[63] J. Magee și D. Boodman, Production Planning and Inventory Control, New York: McGraw-Hill, 1967.

[64] C. Maravelias și C. Sung, „Integration of production planning and scheduling: Overview, challenges and opportunities," *Comput. Chem. Eng.,* vol. 33, nr. 12, pp. 1919-1930, 2009.

[65] X. Shao, L. Xinyu, L. Gao și C. Zhang, „Integration of process planning and scheduling—A modified genetic algorithm-based approach," *Computers & Operations Research,* vol. 36, nr. 6, pp. 2082-2096, 2009.

[66] L. Yuechang și Y. Fang, „Boost the Integration of Planning and Scheduling: a Heuristics Approach," *Procedia Engineering ,* nr. 29, pp. 3348-3352, 2012.

[67] W. Tsai și Y. Liu, A Framework of Production Planning and Control with Carbon Tax under Industry 4.0, Taoyuan: National Center University, 2018.

[68] Z. Zahedi, „Integrated Batch Production and Maintenance Scheduling to Minimize Total Production and Maintenance Costs with a Common Due Date Constraint," în *Industrial Engineering*, New Delhi, New Age International Publishers, 2019.

[69] J. Lohmer și R. Lasch, „Production planning and scheduling in multi-factory production networks: a systematic literature review," *International Journal of Production Research,* vol. 59, nr. 7, pp. 2028-2054, 2021.

[70] T. Roman , „Managementul producției și operațiilor", Iași: suport de curs, 2013.

[71] A. Tamilarasi și A. Kumar, „An enhanced genetic algorithm with simulated annealing for job shop scheduling," *International Journal of Engineering ,* vol. 2, nr. 1, pp. 144-151, 2010.

[72] A. Snatkin și T. Eiskop, „Production monitoring system development and modification," *Proceedings of the Estonian Academy of Sciences,* vol. 64, nr. 4, p. 567, 2015.

[73] B. Chandrayan , „IoT Integration in Industry - A Literature Review," în *Recent Advances in Mechanical Engineering* , New York, Springer, 2020, pp. 9-17 .

[74] C. Blum , „Ant colony optimization: Introduction and recent trends," *Physics of Life Reviews,* vol. 2, nr. 4, pp. 353-373, 2005.

[75] M. Dorigo și M. Brittari, Ant Colony Optimization and swarm intelligence, New York: Springer, 2004.

[76] E. Florez și W. Gomez, „Ant Colony Optimization Algorithm for Job Shop Scheduling Problem," *International Journal of Artifician Intelligence & Application ,* vol. 4, nr. 4, 2013.

[77] L. Wang și G. Zhou, „An effective artificial bee colony algorithm for the flexible job-shop scheduling problem," *The International Journal of Advanced Manufacturing Technology,* vol. 60, pp. 303-315, 2012.

[78] V. Eswaramurthy și A. Tamilarasi, „Hybridizing tabu search with ant colony optimization for solving job shop scheduling problems," *International Journal of Advanced Manufacturing ,* vol. 40, nr. 9-10, pp. 1004-1015, 2010.

[79] E. Bonabeau, M. Dorigo și T. Theraulaz, From Natural to artificial Swarm Intelligence, New York: Oxford University Press, 1999 .

[80] W. Liu, Z. Wang, X. Liu, N. Zeng, Y. Liu și F. Alsaadi, „A survey of deep neural network arhitectures and their application," *Neurocomputing ,* vol. 234, pp. 11-26, 2017.

[81] A. Szeberényi, S. Garg și M. Alqudah, „The Impact of Artificial Intelligence of Management Productivity and Efficiency," *Business Management and Economics,* vol. 22, nr. 01, pp. 424-434, 2024.

[82] D. Ogundipe, S. Badatunde și E. Abaku, „AI and product management: A theoretical overview from idea to market," *International Journal of Management & Entrepreneurship Research,* vol. 6, nr. 3, pp. 950-969, 2024.

[83] P. Burggräf, J. Wagner și B. Heinbach, „Artificial Intelligence in Production Management - A review of the current state of affairs and research trends in academia," în *International Conference on Information Management and Processing (ICIMP 2018*, London, 2018.

[84] N. Sfetcu, „Ciclul de viață al inteligenței artificiale," *IT & C,* vol. 1, nr. 2, pp. 10-26, 2022.

[85] P. Mahajan, „Artificial Intelligence in Product Management," *International Journal of Computer Trends and Technology,* vol. 72, nr. 6, pp. 84-93, 2024.

[86] H. Askarifard, „Types of classifier in artificial intelligence," *International Journal of Computers & Technology ,* vol. 15, nr. 1, pp. 6436-6443, 2016.

[87] T. Fitria, „Artificial Intelligence (AI) In Education: Using AI Tools for Teaching and Learning Process," în *Prosiding Seminar Nasional & Call for Paper STIE AAS*, Surakarta, 2021.

[88] I. Sarker, „Machine Learning: Algorithms, Real-World Applications and Research Directions," *SN Computer Science,* vol. 2, nr. 160, 2021.

[89] J. Cacavid, S. Lamouri, B. Grabot și A. Fortin, „Machine Learning in Production Planning and Control: A Review of Empirical Literature," *Elsevier - Science Direct - IFAC-PapersOnLine,* vol. 52, nr. 13, pp. 385-390, 2019.

[90] A. Abtahi și A. Bijari, „A novel hybrid meta-heuristic technique applied to the well-known benchmark optimization problems," *Journal of Industrial Engineering International,* vol. 13, pp. 93-105, 2017.

[91] B. Jarboui și P. Siarry, Metaheuristics for Production Scheduling (Automation - Control and Industrial Engineering), New York: Willey, 2013.

[92] J. Kennedy și R. Eberhart, Swarm Intellingence, San Diego: Morgan Kaufmann Publishers, 2001.

[93] Y. Li, Z. Hao și H. Lei, „A hybrid artificial bee colony algorithm for flexible job shop schedulling problem," *International Journal of Computers, Communications & Control,* vol. IV, nr. 2, pp. 286-297, 2011.

[94] H. Boukef, M. Benrejeb și P. Borne , „Flexible Job-shop Scheduling Problems Resolution Inspired from Particle Swarm Optimization," *Studies in Informatics and Control,* vol. 17, nr. 3, pp. 241-252, 2008.

[95] K. Gao, P. Suganthan, K. Pan și T. Chua, „An improved artificial bee colony algorithm for flexible job-shop scheduling problem with fuzzy processing time," *Expert Systems with Applications: An International Journal,,* vol. 65, nr. c, pp. 52-67, 2016.

[96] Z. Guo , W. Wong, S. Leung și J. Fan, „Intelligent production control decision support system for flexible assembly lines," *Expert Systems with Applications,* vol. 36, nr. 3, 2009.

[97] D. Lucke, C. Constantinescu și E. Westkamper, „Smart Factory - A Step towards the Next Generation of Manufacturing," în *Manufacturing Systems and Technologies for the New Frontier:* , Tokyo , The 41st CIRP Conference on Manufacturing Systems May 26–28, 2008, pp. 115-118.

[98] P. Pongchairerks și V. Kachitvichyanukul, „A particle swarm optimization algorithm on job-shop scheduling problems with multi-purpose machines," *Asia-Pacific Journal of Operational Research,* vol. 26, nr. 2, pp. 161-184, 2009.

[99] P. Korytkowski, S. Rymaszewski și T. Wisniewski, „Ant Colony Optimization for job scheduling using multi-attribute dispatching rules," *The International Journal of Advanced Manufactoring Technology ,* vol. 67, nr. 1, 2013.

[100] R. Nakandhrakumar și M. Balachandar , „Implementation of Simulated Annealing Technique for Optimizing Job Shop Scheduling Problem," *International Journal of Advanced Mechanical Engineering,* vol. 4, nr. 2, pp. 169-174, 2014.

[101] R. Marian, L. Luong și R. Akararungruangkul, „Optimisation of distribution networks using Genetic Algorithms. Part 2 - The Genetic Algorithm and Genetic Operators," *International Journal of Manufacturing Technology and Management,* vol. 15, nr. 1, pp. 84-101, 2008.

[102] U. Umar și N. Ismail , „Hybrid multiobjective genetic algorithms for integrated dynamic scheduling and routing of jobs and automated-guided vehicle (AGV) in flexible manufacturing systems (FMS) environment," *The International Journal of Advanced Manufacturing Technology,* vol. 81, nr. 9, 2015.

[103] X. Qiu și H. Lau, „An AIS-based hybrid algorithm for static job shop scheduling," *Journal of Intelligent Manufacturing ,* vol. 25, nr. 3, 2014.

[104] T. Jianchao, G. Zhang, B. Lin și B. Zhang, „A Hybrid Algorithm for Flexible Job-Shop Scheduling Problem," *Elsevier - Procedia Engineering,* vol. 15, pp. 3678-3683, 2011.

[105] S. Xiaoyu, Y. Cao și C. Chang, „A Hybrid Algorithm of PSO and SA for Solving JSP," în *Fifth International Conference on Fuzzy Systems and Knowledge Discovery*, 2008.

[106] L. Liu, X. Hu, G. Zhao și Wang S., „A hybrid PSO-GA Algorithm for job shop scheduling in machine tool production," *International Journal of Production Research,* vol. 53, nr. 19, pp. 5751-5781, 2015.

[107] R. Vlad, A. Marton, D. Jucan și M. Bogdan, „Applying Linear Programming to Flow Shop Scheduling," *Management and Economic Engineering,* vol. 14, nr. 4, pp. 880-893, 2015.

[108] A. Guinet, C. Dupuy și V. Botta-Genoulaz, „Batch dispersion model to optimise traceability in food industry," *Journal of Food Engineering,* vol. 70, pp. 333-339, 2005.

[109] S. Plathottam, A. Rzonca, R. Lakhnori și C. Iloeje , „A review of artificial intelligence applications in manufacturing operations," *Journal of Advanced Manufacturing and Processing,* vol. 5, nr. 3, 2023.

[110] K. Bley și S. Fredriksen, „The Role of Organizational Culture on Artificial Intelligence Capabilities and Organizational Performance," în *The Role of Digital Technologies in Shaping the Post-Pandemic World*, New York, Springer, 2022, pp. 13-24.

[111] S. Balasubramanian, „Integration of Artificial Intelligence in the Manufacturing Sector: A Systematic Review of Applications and Implications," *INTERNATIONAL JOURNAL OF PRODUCTION TECHNOLOGY AND MANAGEMENT,* vol. 14, nr. 1, pp. 1-11, 2023.

[112] S. Chaudhuri și L. Krishnan, „Impact of using ai in manufacturing industries," *Journal of the International Academy for Case Studies ,* vol. 28, nr. S4, pp. 1-10, 2022.

[113] X. Jiang , J. Liu, M. Shi și Y. Yang, „Impact of Artificial Intelligence on Manufacturing Industry Global Value Chain Position," *MDPI- Sustainability ,* vol. 16, nr. 3, p. 1341, 2024.

[114] K. Yang-Suk, E. Byun și E. Deelman, „Pegasus on the virtual grid: A case study of workflow planning over captive resources," în *Workflows in Support of Large-Scale Science*, IEEE Xplore, 2008.

[115] K. Lee, N. Paton, R. Sakellariou, E. Deelman, A. Fernandes și G. Mehta, „Adaptive workflow processing and execution in Pegasus," *Wiley Online Library: Concurrency and Computation: Practice and Experience,* vol. 21, nr. 16, pp. 1965-1981, 2009.

[116] E. Valencia, S. Lamouri, R. Pellerin și A. Moeuf, „Modeling of the Master Production Schedule for the Digital Transition of Manufacturing SMEs in the Context of Industry 4.0," *MDPI- Sustainabily,* vol. 14, nr. 19, p. 12562, 2022.

[117] M. Chávez, M. Rangel și M. Cruz-Rosales, „Accelerated simulated annealing algorithm applied to the flexible job shop scheduling problem,” *International Transactions in Operational Research,* vol. 24, nr. 5, 2015.

[118] E. Deelman, K. Vahi și M. Rynge, „The Evolution of the Pegasus Workflow Management Software,” *Computing in Science & Engineering ,* vol. 21, nr. 4, pp. 99-110, 2019.

[119] M. Bailon, R. Espino și J. Flores, „Improvement for Production Management and Control Using Lean Manufacturing Tools in the Manufacturing of Posts and Accessories,” în *Human Interaction, Emerging Technologies and Future Applications III* , New York, Springer, 2021, pp. 560-565.

[120] S. Sumathi, T. Hamsapriya și P. Surekha, Evolutionary Intelligence: An Introduction to Theory and Applications with Matlab, New York: Springer, 2008.

[121] I. -. C. O. S. Documentation, „https://www.ibm.com/docs/en/icos/20.1.0?topic=optimization-examples-multiobjective,” [Interactiv]. [Accesat 4 Decembrie 2023].

[122] R. Bixby, „A Brief History of Linear and Mixed-Integer Programming Computation,” în *Computer Science, History, Mathematics*, Documenta Mathematica , 2012, pp. 107-120.

[123] MathWorks, „https://www.mathworks.com/help/gads/problem-based-setup.html?s_tid=srchbrcm,” [Interactiv]. Available: https://www.mathworks.com/help/gads/problem-based-setup.html?s_tid=srchbrcm. [Accesat 24 Februarie 2024].

LISTA FIGURILOR

LISTA TABELELOR

ANEXA 1. Ghidul de interviu și răspunsurile la întrebări

1. Sunteți interesați să folosiți inteligența IA în optimizarea programării producției?

2. În ce măsură considerați că ar fi acceptat gradul de implementarea IA de către angajații companiei.

3. Care este opinia dvs cu privire la investiția în dezvoltarea softurilor care folosesc IA în programarea producției?

4. Utilizați algoritmi de IA în programarea producției în acest moment (la momentul actual)?

5. Ce capacități/abilități trebuie să aibă angajații care lucrează pentru implementarea IA în programarea producției?

6. Care sunt factorii care facilitează implementarea IA în programarea producției în compania în care lucrați?

7. Considerati că la momentul actual putem vorbi de o eficiență semnificativ îmbunătățită inclusiv reducerea timpilor de așteptare prin utilizarea de IA în managementul producției de serie?

8. Care sunt factorii care împiedică/încetinesc implementarea IA în programarea producției în compania în care lucrați?

9. Cum ați descrie relația pe care o aveți/ați avea-o cu managerii din celorlalte departamente în ceea ce privește implementarea IA în programarea producției?

GDPR

Sunteți de acord să vi se facă publice datele cu caracter public (nume/prenume/companie) ?

Da/Nu

INTERVIU GHIDAT nr. 1:

1. **Sunteți interesați să folosiți inteligența IA în optimizarea programării producției?**

Răspuns: Da. Această optimizare este foarte benefică și ne-ar ajuta să eficientizăm producția.

2. **În ce măsură considerați că ar fi acceptat gradul de implementare a IA de către angajații companiei?**

Răspuns: Angajații sunt foarte deschiși la implementarea IA în companie, demonstrând o atitudine pozitivă.

3. **Care este opinia dvs. cu privire la investiția în dezvoltarea softurilor care folosesc IA în programarea producției?**

Răspuns: Părerea mea este una pozitivă, deoarece gradul de eroare este mult redus și crește eficiența.

4. **Utilizați algoritmi de IA în programarea producției în acest moment?**

Răspuns: Da. Folosim IA în acest moment în programarea producției pentru a îmbunătăți procesele.

5. **Ce capacități/abilități trebuie să aibă angajații care lucrează pentru implementarea IA în programarea producției?**

Răspuns: Angajații trebuie să aibă cunoștințe solide de operare a calculatorului și înțelegerea algoritmilor de IA.

6. **Care sunt factorii care facilitează implementarea IA în programarea producției în compania în care lucrați?**

Răspuns: Factorii sunt: creșterea productivității, reducerea erorilor și optimizarea proceselor tehnologice.

7. **Considerați că la momentul actual putem vorbi de o eficiență semnificativ îmbunătățită, inclusiv reducerea timpilor de așteptare, prin utilizarea de IA în managementul producției de serie?**

Răspuns: Da. Implementarea IA este cea mai bună și eficientă metodă de îmbunătățire a timpilor de așteptare.

8. **Care sunt factorii care împiedică/încetinesc implementarea IA în programarea producției în compania în care lucrați?**

Răspuns: În compania noastră nu toate procesele se pretează pentru a fi coordonate de IA, ceea ce încetinește implementarea.

9. **Cum ați descrie relația pe care o aveți/ați avea-o cu managerii din celelalte departamente în ceea ce privește implementarea IA în programarea producției?**

Răspuns: Toți suntem de acord și deschiși pentru a îmbunătăți și eficientiza procesele tehnologice. Avem o relație de colaborare și susținem implementarea IA.

Nume: Și-a păstrat anonimatul sub sancțiunea normelor GDPR Compania LEONI Wiring System Bistrita

INTERVIU GHIDAT nr. 2:

1. **Sunteți interesați să folosiți inteligența IA în optimizarea programării producției?**

Răspuns: Nu, în acest moment nu suntem interesați.

2. **În ce măsură considerați că ar fi acceptat gradul de implementare a IA de către angajații companiei?**

Răspuns: Depinde de angajat și de cât de obișnuit este acesta cu tehnologia în general. Gradul de acceptare variază.

3. **Care este opinia dvs. cu privire la investiția în dezvoltarea softurilor care folosesc IA în programarea producției?**

Răspuns: Consider că este oportună, însă depinde mult de raportul câștig/cost și de beneficiile concrete.

4. **Utilizați algoritmi de IA în programarea producției în acest moment?**

Răspuns: Nu, nu utilizăm algoritmi de IA în acest moment.

5. **Ce capacități/abilități trebuie să aibă angajații care lucrează pentru implementarea IA în programarea producției?**

Răspuns: Nu știu exact. Ar trebui să aibă, probabil, cunoștințe tehnice și de operare a calculatorului.

6. **Care sunt factorii care facilitează implementarea IA în programarea producției în compania în care lucrați?**

Răspuns: Utilizarea facilă, capacitatea de a face analize pertinente și costurile sunt factori importanți.

7. **Considerați că la momentul actual putem vorbi de o eficiență semnificativ îmbunătățită, inclusiv reducerea timpilor de așteptare, prin utilizarea de IA în managementul producției de serie?**

Răspuns: Depinde de industrie. În unele industrii, IA poate aduce îmbunătățiri semnificative.

8. **Care sunt factorii care împiedică/încetinesc implementarea IA în programarea producției în compania în care lucrați?**

Răspuns: Nu am văzut oportunitatea în acest moment. Nu toate procesele noastre se pretează pentru IA.

9. **Cum ați descrie relația pe care o aveți/ați avea-o cu managerii din celelalte departamente în ceea ce privește implementarea IA în programarea producției?**

Răspuns: Avem o colaborare bună în toate fazele implementării și suntem deschiși la îmbunătățiri.

Nume: Și-a păstrat anonimatul sub sancțiunea normelor GDPR - Compania: Michelin Zalău

INTERVIU GHIDAT nr. 3:

1. Sunteți interesați să folosiți inteligența IA în optimizarea programării producției?

Răspuns: Suntem parțial interesați deoarece dorim să înțelegem mai bine beneficiile și să evaluăm costurile și impactul asupra proceselor noastre actuale. De asemenea, avem nevoie de o perioadă de testare pentru a ne asigura că aceste tehnologii se integrează bine cu sistemele noastre existente.

2. În ce măsură considerați că ar fi acceptat gradul de implementare IA de către angajații companiei?

Răspuns: Gradul de acceptare ar fi redus, din cauza temerilor legate de schimbare și a lipsei de familiaritate cu tehnologiile noi. Angajații ar putea fi reticenți în a adopta IA din cauza preocupărilor legate de securitatea locului de muncă și de încrederea în noile tehnologii.

3. Care este opinia dvs cu privire la investiția în dezvoltarea softurilor care folosesc IA în programarea producției?

Răspuns: Investiția poate fi benefică sau nu, depinde de fiecare mediu de producție și activitatea prestată, deoarece costurile și beneficiile variază semnificativ. Trebuie efectuată o analiză cost-beneficiu detaliată pentru a determina dacă investiția va aduce îmbunătățiri semnificative și sustenabile.

4. Utilizați algoritmi de IA în programarea producției în acest moment (la momentul actual)?

Răspuns: Nu, nu utilizăm algoritmi de IA în programarea producției în acest moment. Încă ne bazăm pe metode tradiționale și euristice pentru a optimiza procesele de producție.

5. Ce capacități/abilități trebuie să aibă angajații care lucrează pentru implementarea IA în programarea producției?

Răspuns: Nu știu exact ce capacități/abilități sunt necesare, dar presupun că ar fi nevoie de cunoștințe solide în domeniul IT, înțelegerea algoritmilor de IA, abilitatea de a analiza și interpreta datele, și competențe în gestionarea proiectelor tehnologice

6. Care sunt factorii care facilitează implementarea IA în programarea producției în compania în care lucrați?

Răspuns: Infrastructura adecvată, angajamentul organizațional, comunicarea eficientă între departamente, securitatea și confidențialitatea datelor, precum și flexibilitatea și adaptabilitatea organizației sunt factori esențiali pentru facilitarea implementării IA în programarea producției.

7. Considerați că la momentul actual putem vorbi de o eficiență semnificativ îmbunătățită, inclusiv reducerea timpilor de așteptare prin utilizarea de IA în managementul producției de serie?

Răspuns: Nu știu sigur dacă putem vorbi deja de o eficiență semnificativ îmbunătățită și reducerea timpilor de așteptare prin utilizarea de IA, deoarece încă nu avem experiența necesară și datele concrete care să demonstreze acest lucru în contextul nostru specific.

8. Care sunt factorii care împiedică/încetinesc implementarea IA în programarea producției în compania în care lucrați?

Răspuns: Infrastructura IT inadecvată, lipsa expertizei tehnice necesare, costurile ridicate asociate cu implementarea și mentenanța sistemelor de IA, precum și rezistența la schimbare din partea angajaților sunt principalii factori care împiedică sau încetinesc implementarea IA.

9. Cum ați descrie relația pe care o aveți/ați avea-o cu managerii din celelalte departamente în ceea ce privește implementarea IA în programarea producției?

Răspuns: Relația ar trebui să fie una de colaborare strânsă și comunicare constantă, pentru a asigura o implementare eficientă a IA. Este esențial ca toți managerii să fie aliniați la obiectivele comune și să susțină eforturile de integrare a noilor tehnologii pentru a depăși eventualele obstacole și a maximiza beneficiile.

Nume: Și-a păstrat anonimatul sub sancțiunea normelor GDPR

Compania: ABC

INTERVIU GHIDAT nr. 4:

1. Sunteți interesați să folosiți inteligența IA în optimizarea programării producției?

Răspuns: Momentan nu, nu ne situăm într-o fază pregătită pentru IA încă în organizația noastră. Considerăm că trebuie să ne consolidăm procesele de bază înainte de a adopta tehnologii avansate.

2. În ce măsură considerați că ar fi acceptat gradul de implementare IA de către angajații companiei?

Răspuns: Angajații ar accepta dacă s-ar explica beneficiile în faza inițială și dacă pe parcursul implementării ar fi actualizați constant. Este crucial să existe o comunicare clară și transparentă pentru a reduce temerile și reticențele.

3. Care este opinia dvs cu privire la investiția în dezvoltarea softurilor care folosesc IA în programarea producției?

Răspuns: Momentan există o problemă reală în a găsi un dezvoltator de soft capabil în România care să dezvolte un modul de programare suficient de „smart”. Modulele de APS (Advanced Planning System) existente sunt rigide, iar furnizorii acestora nu reușesc implementarea astfel încât să fie mulată pe nevoile clientului, ca acesta să beneficieze cu adevărat de un instrument automatizat, venit în ajutorul persoanei care e responsabilă de planificarea producției.

4. Utilizați algoritmi de IA în programarea producției în acest moment (la momentul actual)?

Răspuns: Nu, nu utilizăm algoritmi de IA în programarea producției în acest moment.

5. Ce capacități/abilități trebuie să aibă angajații care lucrează pentru implementarea IA în programarea producției?

Răspuns: Angajații trebuie să aibă o înțelegere a fluxurilor din producție, cunoașterea produselor finite și o înțelegere generală asupra gestiunilor (de la materii prime până la produse finite). Este esențial să poată judeca când și cum să intervină în modelarea producției cu propria decizie bazată pe o înțelegere contextuala potrivită înainte de a lua o decizie de schimbare sau modificare.

6. Care sunt factorii care facilitează implementarea IA în programarea producției în compania în care lucrați?

Răspuns: Nu există contextul potrivit pentru moment pentru a implementa IA. Ne concentrăm pe consolidarea infrastructurii și a proceselor de bază înainte de a considera implementarea IA.

7. Considerați că la momentul actual putem vorbi de o eficiență semnificativ îmbunătățită, inclusiv reducerea timpilor de așteptare prin utilizarea de IA în managementul producției de serie?

Răspuns: În cazul unei implementări de succes, clar da, valoarea stocurilor ar fi mult mai mică dacă ar exista o modelare mai complexă care să țină cont de toți factorii implicați (începând de la comenzi de materii prime până la livrare). Cea mai importantă și dăunătoare pierdere pentru producție este supraproducția; cu siguranță șansele de a ține sub control acest aspect ar fi crescute exponential dacă dezvoltarea IA și implementarea ar fi de succes în prealabil.

8. Care sunt factorii care împiedică/încetinesc implementarea IA în programarea producției în compania în care lucrați?

Răspuns: În momentul de față, suntem în faza de creare a structurilor necesare (revizuire rețete produs, revizuire routing-uri pentru semifabricate, normare timpi de producție pe utilaje, revalidare timpi de manoperă și timpi mașină). Odată cu finalizarea acestei faze, trebuie înțeles modelul de business și tipologia clienților din punct de vedere al plasării comenzilor (periodicitate, împărțire de strangers-repeaters-runners-high runners), înțelegerea input-ului pentru această modelare, respectiv răspunsul la întrebări precum: se lucrează după un istoric al comenzilor? Se lucrează după un forecast care are la bază istoric + coeficienți de ajustare? Există leadtime-uri diferite pentru clienți diferiți? Eficacitatea echipamentelor din producție este măsurată și credibilă? Care este rata de rebut pe fiecare fază de producție? Sunt mulți parametri diferiți și fiecare are propria pondere de importanță în această programare.

9. Cum ați descrie relația pe care o aveți/ați avea-o cu managerii din celelalte departamente în ceea ce privește implementarea IA în programarea producției?

Răspuns: Cred că ar exista deschidere, atât timp cât se prezintă beneficiile într-un mod structurat și se oferă exemple relevante ale furnizorului de soft de la alte implementări de succes. Este important ca toți membrii echipei de management să-și asume de la bun început că pe parcurs vor fi multe dileme și situații care vor trebui soluționate „în cel mai bun mod posibil în acel moment”. Soluția perfectă nu există, dar este esențial să îmbunătățim continuu, lucrând împreună.

Nume: Chesoan Robert

Compania: Savini Due SRL

GDPR

Sunteți de acord să vi se facă publice datele cu caracter public (nume/prenume/companie) ?

Da/Nu

INTERVIU GHIDAT nr. 5

1. Sunteți interesați să folosiți inteligența artificială în optimizarea programării producției?

Răspuns: Da, suntem interesați deoarece IA poate aduce îmbunătățiri semnificative în eficiență și productivitate.

2. În ce măsură considerați că ar fi acceptat gradul de implementare a IA de către angajații companiei?

Răspuns: Acceptarea ar fi destul de mare, mai ales dacă beneficiile sunt explicate clar și angajații sunt actualizați constant pe parcursul implementării.

3. Care este opinia dvs. cu privire la investiția în dezvoltarea softurilor care folosesc IA în programarea producției?

Răspuns: Investiția în IA este crucială pentru a automatiza procesele, reduce costurile și îmbunătăți eficiența, productivitatea și calitatea produselor în mediul competitiv actual.

4. Utilizați algoritmi de IA în programarea producției în acest moment (la momentul actual)?

Răspuns: Nu, momentan nu folosim algoritmi de IA pentru programarea producției.

5. Ce capacități/abilități trebuie să aibă angajații care lucrează pentru implementarea IA în programarea producției?

Răspuns: Angajații trebuie să aibă abilități de programare și capacitatea de a înțelege procesul de producție.

6. Care sunt factorii care facilitează implementarea IA în programarea producției în compania în care lucrați?

Răspuns: Existența datelor necesare pentru programarea producției în sistemele informatice actuale.

7. Considerați că la momentul actual putem vorbi de o eficiență semnificativ îmbunătățită, inclusiv reducerea timpilor de așteptare, prin utilizarea de IA în managementul producției de serie?

Răspuns: În cazul unei implementări de succes, clar da, valoarea stocurilor ar fi mult mai mică dacă ar exista o modelare complexă care să țină cont de toți factorii implicați.

8. Care sunt factorii care împiedică/încetinesc implementarea IA în programarea producției în compania în care lucrați?

Răspuns: Lipsa de timp și resurse necesare pentru implementare.

9. Cum ați descrie relația pe care o aveți/ați avea-o cu managerii din celelalte departamente în ceea ce privește implementarea IA în programarea producției?

Răspuns: Avem o relație de colaborare și de lucru în echipă pentru implementarea IA, bazată pe comunicare și obiective comune.

Nume: Și-a păstrat anonimatul sub sancțiunea normelor GDPR

Compania: Verdino Green Foods

INTERVIU GHIDAT nr. 6

1. Sunteți interesați să folosiți inteligența artificială în optimizarea programării producției?

Răspuns: Da, suntem interesați deoarece IA poate aduce îmbunătățiri semnificative în eficiență și adaptabilitate.

2. În ce măsură considerați că ar fi acceptat gradul de implementare a IA de către angajații companiei?

Răspuns: Acceptarea ar fi moderată, depinde de cum sunt prezentate beneficiile și de instruirea oferită.

3. Care este opinia dvs. cu privire la investiția în dezvoltarea softurilor care folosesc IA în programarea producției?

Răspuns: Investiția este necesară pentru a automatiza procesele și a îmbunătăți eficiența și productivitatea.

4. Utilizați algoritmi de IA în programarea producției în acest moment (la momentul actual)?

Răspuns: Nu, momentan nu folosim algoritmi de IA în programarea producției.

5. Ce capacități/abilități trebuie să aibă angajații care lucrează pentru implementarea IA în programarea producției?

Răspuns: Angajații trebuie să aibă abilități de programare și o bună înțelegere a procesului de producție.

6. Care sunt factorii care facilitează implementarea IA în programarea producției în compania în care lucrați?

Răspuns: Existența unei infrastructuri IT solide și a unor politici de suport clare.

7. Considerați că la momentul actual putem vorbi de o eficiență semnificativ îmbunătățită, inclusiv reducerea timpilor de așteptare, prin utilizarea de IA în managementul producției de serie?

Răspuns: Nu, încă nu avem dovezi clare că IA poate reduce semnificativ timpii de așteptare.

8. Care sunt factorii care împiedică/încetinesc implementarea IA în programarea producției în compania în care lucrați?

Răspuns: Lipsa resurselor și a timpului necesar pentru implementare.

9. Cum ați descrie relația pe care o aveți/ați avea-o cu managerii din celelalte departamente în ceea ce privește implementarea IA în programarea producției?

Răspuns: Relația ar fi una de colaborare și comunicare constantă.

Nume: Și-a păstrat anonimatul sub sancțiunea normelor GDPR

Compania: LANTIC

INTERVIU GHIDAT nr. 7

1. Sunteți interesați să folosiți inteligența artificială în optimizarea programării producției?

Răspuns: Da, considerăm IA esențială pentru îmbunătățirea performanței și a eficienței.

2. În ce măsură considerați că ar fi acceptat gradul de implementare a IA de către angajații companiei?

Răspuns: Acceptarea ar fi destul de mare, dacă beneficiile sunt explicate clar și suportul este asigurat.

3. Care este opinia dvs. cu privire la investiția în dezvoltarea softurilor care folosesc IA în programarea producției?

Răspuns: Investiția este crucială pentru a automatiza procesele și a reduce costurile operaționale.

4. Utilizați algoritmi de IA în programarea producției în acest moment (la momentul actual)?

Răspuns: Nu, nu utilizăm algoritmi de IA în acest moment.

5. Ce capacități/abilități trebuie să aibă angajații care lucrează pentru implementarea IA în programarea producției?

Răspuns: Abilități de programare, cunoștințe solide de producție și deschidere la noi tehnologii.

6. Care sunt factorii care facilitează implementarea IA în programarea producției în compania în care lucrați?

Răspuns: Suportul managementului și accesul la resurse tehnologice adecvate.

7. Considerați că la momentul actual putem vorbi de o eficiență semnificativ îmbunătățită, inclusiv reducerea timpilor de așteptare, prin utilizarea de IA în managementul producției de serie?

Răspuns: Nu, avem nevoie de mai multe date pentru a confirma această ipoteză.

8. Care sunt factorii care împiedică/încetinesc implementarea IA în programarea producției în compania în care lucrați?

Răspuns: Resurse limitate și lipsa unei infrastructuri IT adecvate.

9. Cum ați descrie relația pe care o aveți/ați avea-o cu managerii din celelalte departamente în ceea ce privește implementarea IA în programarea producției?

Răspuns: Relația este bazată pe colaborare și suport reciproc.

Nume: Și-a păstrat anonimatul sub sancțiunea normelor GDPR

Compania: Pehart Tec

INTERVIU GHIDAT nr. 8

1. Sunteți interesați să folosiți inteligența artificială în optimizarea programării producției?

Răspuns: Da, suntem deschiși să folosim IA pentru a crește eficiența și a reduce costurile.

2. În ce măsură considerați că ar fi acceptat gradul de implementare a IA de către angajații companiei?

Răspuns: Acceptarea ar fi mare, dacă angajații sunt bine informați și instruiți.

3. Care este opinia dvs. cu privire la investiția în dezvoltarea softurilor care folosesc IA în programarea producției?

Răspuns: Este o investiție necesară pentru a menține competitivitatea și a optimiza procesele.

4. Utilizați algoritmi de IA în programarea producției în acest moment (la momentul actual)?

Răspuns: Nu, în prezent nu utilizăm astfel de algoritmi.

5. Ce capacități/abilități trebuie să aibă angajații care lucrează pentru implementarea IA în programarea producției?

Răspuns: Abilități tehnice, înțelegere a proceselor de producție și adaptabilitate.

6. Care sunt factorii care facilitează implementarea IA în programarea producției în compania în care lucrați?

Răspuns: Suport managerial și o infrastructură tehnologică solidă.

7. Considerați că la momentul actual putem vorbi de o eficiență semnificativ îmbunătățită, inclusiv reducerea timpilor de așteptare, prin utilizarea de IA în managementul producției de serie?

Răspuns: Nu, încă nu putem vorbi de o eficiență îmbunătățită fără date concrete.

8. Care sunt factorii care împiedică/încetinesc implementarea IA în programarea producției în compania în care lucrați?

Răspuns: Lipsa de timp, resurse și infrastructură tehnologică adecvată.

9. Cum ați descrie relația pe care o aveți/ați avea-o cu managerii din celelalte departamente în ceea ce privește implementarea IA în programarea producției?

Răspuns: Relația este una de colaborare strânsă și suport reciproc.

Nume: Și-a păstrat anonimatul sub sancțiunea normelor GDPR

Compania: Ciserom

INTERVIU GHIDAT NR 9.

1. Sunteți interesați să folosiți inteligența artificială în optimizarea programării producției?

Răspuns: Da, credem că IA poate optimiza considerabil programarea producției.

2. În ce măsură considerați că ar fi acceptat gradul de implementare a IA de către angajații companiei?

Răspuns: Acceptarea ar fi ridicată dacă se oferă formare adecvată și beneficii clare.

3.Care este opinia dvs. cu privire la investiția în dezvoltarea softurilor care folosesc IA în programarea producției?

Răspuns: Este o investiție strategică pentru a reduce costurile și a crește eficiența.

4. Utilizați algoritmi de IA în programarea producției în acest moment (la momentul actual)?

Răspuns: Nu, deocamdată nu folosim algoritmi de IA.

5.Ce capacități/abilități trebuie să aibă angajații care lucrează pentru implementarea IA în programarea producției?

Răspuns: Cunoștințe IT, înțelegerea procesului de producție și capacitate de adaptare.

6. Care sunt factorii care facilitează implementarea IA în programarea producției în compania în care lucrați?

Răspuns: Sprijinul managerial și o infrastructură IT adecvată.

7. Considerați că la momentul actual putem vorbi de o eficiență semnificativ îmbunătățită, inclusiv reducerea timpilor de așteptare, prin utilizarea de IA în managementul producției de serie?

Răspuns: Nu, eficiența semnificativ îmbunătățită necesită mai multe date și timp.

8. Care sunt factorii care împiedică/încetinesc implementarea IA în programarea producției în compania în care lucrați?

Răspuns: Resurse limitate și procese de implementare complexe.

9. Cum ați descrie relația pe care o aveți/ați avea-o cu managerii din celelalte departamente în ceea ce privește implementarea IA în programarea producției?

Răspuns: Relația ar fi bazată pe colaborare și comunicare eficientă pentru succesul implementării.

Nume: Și-a păstrat anonimatul sub sancțiunea normelor GDPR

Compania: Kronospan Sebes

INTERVIU GHIDAT NR 10.

1. Sunteți interesați să folosiți inteligența artificială în optimizarea programării producției?

Răspuns: Da, considerăm că IA poate aduce îmbunătățiri semnificative în eficiența și adaptabilitatea programării producției noastre.

2. În ce măsură considerați că ar fi acceptat gradul de implementare a IA de către angajații companiei?

Răspuns: Acceptarea ar fi destul de mare, mai ales dacă beneficiile sunt explicate clar și angajații sunt actualizați constant pe parcursul implementării.

3. Care este opinia dvs. cu privire la investiția în dezvoltarea softurilor care folosesc IA în programarea producției?

Răspuns: Investiția este promițătoare, căutăm soluții care să integreze IA în programarea producției, pornind de la estimarea vânzărilor până la achiziții.

4. Utilizați algoritmi de IA în programarea producției în acest moment (la momentul actual)?

Răspuns: Nu, momentan nu folosim algoritmi de IA pentru programarea producției.

5. Ce capacități/abilități trebuie să aibă angajații care lucrează pentru implementarea IA în programarea producției?

Răspuns: Trebuie să cunoască bine procesul de producție, să fie abili în programarea producției și deschiși la noi sisteme informatice.

6. Care sunt factorii care facilitează implementarea IA în programarea producției în compania în care lucrați?

Răspuns: Existența datelor necesare pentru programarea producției în sistemele informatice actuale.

7. Considerați că la momentul actual putem vorbi de o eficiență semnificativ îmbunătățită, inclusiv reducerea timpilor de așteptare, prin utilizarea de IA în managementul producției de serie?

Răspuns: În cazul unei implementări de succes, clar da, valoarea stocurilor ar fi mult mai mică dacă ar exista o modelare complexă care să țină cont de toți factorii implicați.

8. Care sunt factorii care împiedică/încetinesc implementarea IA în programarea producției în compania în care lucrați?

Răspuns: Lipsa de timp și resurse necesare pentru implementare.

9. Cum ați descrie relația pe care o aveți/ați avea-o cu managerii din celelalte departamente în ceea ce privește implementarea IA în programarea producției?

Răspuns: Managerii sunt de acord cu implementarea unei soluții IA, fiind deschiși la colaborare și la găsirea soluțiilor optime împreună.

Nume: Și-a păstrat anonimatul sub sancțiunea normelor GDPR

Compania: Farmec SA

ANEXA 2. Fișierele .mode și .dat ale modelul codului folosit în CPLEX OPL prin Programare Liniară(MILP).

Fisierul modelului .mod

```
/*********************************************
 * OPL 12.10.0.0 Model
 * Author: Mircea
 * Creation Date: Aug 13, 2024 at 6:52:49 PM
 *********************************************/

 //Tipuri de date necesare pentru citirea marimilor de intrare
```

```
	int nrMaximPrecedente = ...;
	range domeniuPrecedente = 1..nrMaximPrecedente;

	tuple tipComanda {
		string numeComanda;
		string numeProdus;
		int volum;
		int dataIntrare;
		int dataLivrare;
	};
	tuple tipOperatiune {
		int indiceOper;
		string numeComanda;
		string numeOper;
		string utilaj;
		int durata;
	};
	tuple tipPrecedenta {
		int codOper;
		int codPredecesor;
	};

	{string} Comenzi = ...;
	{int} Operatiuni = ...;
	{int} setUtilaje =  ...;

	tipComanda dateComenzi[Comenzi] = ...;
	tipOperatiune dateOperatiuni[Operatiuni] = ...;
	tipPrecedenta setPrecedente[domeniuPrecedente] = ...;

	int k = 10000000;
			//constanta intreaga a carei valoare este aleasa in mod arbitrar

//Variabile de decizie
	dvar float+ T[Operatiuni];
		//momentul la care este programata operatiunea "i"
	dvar boolean Z[Operatiuni][Operatiuni];
		//variabila binara care are valoarea "1" atunci cand operatiunea "i" este
programata
		//inaintea operatiunii "j" pe masina "m"

//	dvar int inceputUtilizare[setUtilaje];
//	dvar int sfarsitUtilizare[setUtilaje];
//
//	dvar int inceputComanda[Comenzi];
//	dvar int sfarsitComanda[Comenzi];


//Functia obiectiv
minimize max(i in Operatiuni) T[i];
//
subject to {

	forall(i in Operatiuni, j in Operatiuni: j != i && dateOperatiuni[i].utilaj ==
dateOperatiuni[j].utilaj)
	CNS1: T[i] - T[j] + k * Z[i][j] >= dateOperatiuni[j].durata;
```

```
	forall(i in Operatiuni, j in Operatiuni: j != i && dateOperatiuni[i].utilaj ==
dateOperatiuni[j].utilaj)
		CNS2: T[j] - T[i] + k * (1 - Z[i][j]) >= dateOperatiuni[i].durata;

	forall(i in domeniuPrecedente)
		CNS3: T[setPrecedente[i].codOper] - T[setPrecedente[i].codPredecesor] >=
dateOperatiuni[setPrecedente[i].codPredecesor].durata;

	forall(i in Operatiuni)
		CNS4: T[i] <= dateComenzi[dateOperatiuni[i].numeComanda].dataLivrare;

	forall(i in Operatiuni)
		CNS5: Z[i][i] == 0;
}

	tuple dateIesire_Operatiuni {
		string Comanda_Nume;
		string Operatiune_Nume;
		float inceput;
		float sfarsit;
		string utilaj;
	};

	setof(dateIesire_Operatiuni) valOptim = { <dateOperatiuni[o].numeComanda,
dateOperatiuni[o].numeOper, T[o],
			T[o] + dateOperatiuni[o].durata, dateOperatiuni[o].utilaj> | o in
Operatiuni };
```

Fisierul datelor .dat

```
/*********************************************
 * OPL 12.10.0.0 Data
 * Author: Mircea
 * Creation Date: Aug 13, 2024 at 6:52:49 PM
 *********************************************/

 SheetConnection Sheet7("Date_EF_MIP_2.xlsm");

Comenzi from SheetRead(Sheet7,"MIP_ListaComenzi");
dateComenzi from SheetRead(Sheet7,"MIP_DateIntrare_Comenzi");

nrMaximPrecedente from SheetRead(Sheet7,"MIP_NumarPrecedente");
setPrecedente from SheetRead(Sheet7,"MIP_Precedente");


//SheetConnection Sheet7("Date_EF_MIP_2.xlsm");
Operatiuni from SheetRead(Sheet7,"MIP_ListaOperatiuni");
dateOperatiuni from SheetRead(Sheet7,"MIP_DateIntrare_Operatiuni");

setUtilaje from SheetRead(Sheet7,"MIP_ListaUtilaje");
//dateAlocari from SheetRead(Sheet7,"LP_Alocari");
```

ANEXA 3. Fișierele .mode și .dat ale modelul codului folosit în CPLEX OPL prin Programare cu Constrângeri(CP).

Fisierul modelului .mod

```
/*********************************************
 * OPL 22.1.1.0 Model
 * Author: Mircea
 * Creation Date: Jul 17, 2024 at 2:50:16 PM
 *********************************************/
using CP;
//Tipuri de date necesare pentru citirea marimilor de intrare
	int nrMaximPrecedente = ...;
	range domeniuPrecedente = 1..nrMaximPrecedente;

	tuple tipComanda {
		string numeComanda;
		string numeProdus;
		int volum;
		int dataIntrare;
		int dataLivrare;
	};
	tuple tipOperatiune {
		int indiceOper;
		string numeComanda;
		string numeOper;
		string utilaj;
		int durata;
	};
	tuple tipPrecedenta {
		string numeOper;
		string numePredecesor;
	};

	{string} Comenzi = ...;
	{string} Operatiuni = ...;
	{string} setUtilaje =  ...;

	tipComanda dateComenzi[Comenzi] = ...;
	tipOperatiune dateOperatiuni[Operatiuni] = ...;
	tipPrecedenta setPrecedente[domeniuPrecedente] = ...;

//Variabile de decizie
	dvar interval intOperatiuni[o in Operatiuni] size  dateOperatiuni[o].durata;
	dvar sequence conflicte[m in setUtilaje]
		in all(o in Operatiuni: dateOperatiuni[o].utilaj == m) intOperatiuni[o];

	dvar int inceputUtilizare[setUtilaje];
	dvar int sfarsitUtilizare[setUtilaje];
```

```
	dvar int inceputComanda[Comenzi];
	dvar int sfarsitComanda[Comenzi];

//Functia obiectiv
minimize max(o in Operatiuni) endOf(intOperatiuni[o]);

//Constrangeri
subject to {
	forall (c in Comenzi, o in Operatiuni: dateComenzi[c].numeComanda ==
dateOperatiuni[o].numeComanda)
		startOf(intOperatiuni[o]) >= dateComenzi[c].dataIntrare;

	forall (c in Comenzi, o in Operatiuni: dateComenzi[c].numeComanda ==
dateOperatiuni[o].numeComanda)
		endOf(intOperatiuni[o]) <= dateComenzi[c].dataLivrare;

	forall (i in domeniuPrecedente) // endBeforeStart startBeforeStart
		endBeforeStart(intOperatiuni[setPrecedente[i].numePredecesor],
					intOperatiuni[setPrecedente[i].numeOper]);

	forall (c in Comenzi)
		inceputComanda[c] == min(o in Operatiuni: dateOperatiuni[o].numeComanda ==
c) startOf(intOperatiuni[o]);

	forall (c in Comenzi)
		sfarsitComanda[c] == max(o in Operatiuni: dateOperatiuni[o].numeComanda ==
c) endOf(intOperatiuni[o]);

//	forall (c in Comenzi)
//		sfarsitComanda[c] <= inceputComanda[c] + 7 * 24 * 3600;

	forall (m in setUtilaje)
		noOverlap(conflicte[m]);

	forall (m in setUtilaje)
		inceputUtilizare[m] == min(o in Operatiuni: dateOperatiuni[o].utilaj == m)
startOf(intOperatiuni[o]);

	forall (m in setUtilaje)
		inceputUtilizare[m] >= 0;

	forall (m in setUtilaje)
		sfarsitUtilizare[m] == max(o in Operatiuni: dateOperatiuni[o].utilaj == m)
endOf(intOperatiuni[o]);
}
	tuple dateIesire_Operatiuni {
		string Comanda_Nume;
		string Operatiune_Nume;
		int inceput;
		int sfarsit;
		string utilaj;
	};

	tuple dateIesire_Utilaje {
		string utilaj;
		int inceput;
```

```
		int sfarsit;
	};

	tuple dateIesire_Comenzi {
		string comanda;
		int inceput;
		int sfarsit;
		int durataTotala;
	};

	setof(dateIesire_Operatiuni) valOptim = { <dateOperatiuni[o].numeComanda,
dateOperatiuni[o].numeOper, startOf(intOperatiuni[o]),
			endOf(intOperatiuni[o]), dateOperatiuni[o].utilaj> | o in Operatiuni
};

	setof(dateIesire_Utilaje) progUtilaje = { <m, inceputUtilizare[m],
sfarsitUtilizare[m]> | m in setUtilaje };

	setof(dateIesire_Comenzi) progComenzi = { <c,
							min(o in Operatiuni:
dateOperatiuni[o].numeComanda == c) startOf(intOperatiuni[o]),
							max(o in Operatiuni:
dateOperatiuni[o].numeComanda == c) endOf(intOperatiuni[o]),
							sum(o in Operatiuni:
dateOperatiuni[o].numeComanda == c) dateOperatiuni[o].durata> | c in Comenzi };
```

Fisierul datelor .dat

```
/*********************************************
 * OPL 22.1.1.0 Data
 * Author: Mircea
 * Creation Date: Jul 17, 2024 at 2:50:16 PM
 *********************************************/
SheetConnection Sheet3("Date_EF.xlsm");

Comenzi from SheetRead(Sheet3,"ListaComenzi");
dateComenzi from SheetRead(Sheet3,"DateIntrare_Comenzi");

Operatiuni from SheetRead(Sheet3,"ListaOperatiuni");
dateOperatiuni from SheetRead(Sheet3,"DateIntrare_Operatiuni");

nrMaximPrecedente from SheetRead(Sheet3,"NumarPrecedente");
setPrecedente from SheetRead(Sheet3,"Precedente");

SheetConnection Sheet1("Date_EF.xlsm");
setUtilaje from SheetRead(Sheet1,"ListaUtilaje");



SheetConnection Sheet4("Date_EF.xlsm");

valOptim to SheetWrite(Sheet4, "Opl_ProgOperatiuni");
progUtilaje to SheetWrite(Sheet4, "Opl_ProgUtilaje");
progComenzi to SheetWrite(Sheet4, "Opl_ProgComenzi");
```

ANEXA 4. Script Visual Basic pentru aranjarea datelor în vederea folosirii lor de către modelul creat in CPLEX OPL CP

```
Dim Zona_NormeTimp As Range
Dim Zona_ProceseTeh As Range
Dim ZonaCitire_Comenzi As Range
Dim ZonaScriere_Comenzi As Range
Dim Zona_Operatiuni As Range
Dim Zona_Produse As Range
Dim Zona_Precedente As Range

Private Sub Worksheet_SelectionChange(ByVal Target As Range)
   Set ZonaCitire_Comenzi = Workbooks("Date_EF.xlsm").Names("DateInitiale_Comenzi").RefersToRange
   Set ZonaScriere_Comenzi =
Workbooks("Date_EF.xlsm").Names("DateIntrare_Comenzi").RefersToRange
   Set Zona_NormeTimp = Workbooks("Date_EF.xlsm").Names("NormeTimp").RefersToRange
   Set Zona_ProceseTeh = Workbooks("Date_EF.xlsm").Names("DateIntrare_Procese").RefersToRange

   Set Zona_Operatiuni = Workbooks("Date_EF.xlsm").Names("DateIntrare_Operatiuni").RefersToRange
   Set Zona_Produse = Workbooks("Date_EF.xlsm").Names("Lista_Produse").RefersToRange
   Set Zona_Precedente = Workbooks("Date_EF.xlsm").Names("Precedente").RefersToRange

   Dim ZonaClick As Range
   Set ZonaClick = Range("A1")

   If Not Application.Intersect(ZonaClick, Range(Target.Address)) Is Nothing Then
      'Call GenerareDateIntrare_Comenzi
      MsgBox ("Datele comenzilor au fost generate!")

      'Call GenerareDateIntrare_ProcesTehnologic
      Call GenerareDateIntrare_Operatiuni
      MsgBox ("Lista de operatiuni de programat a fost generata!")

      'Call GenerareLista_Precedente
      MsgBox ("Lista de precedente a fost generata!")
   End If
End Sub

Public Sub GenerareDateIntrare_Comenzi()
   ZonaScriere_Comenzi.Clear

   Dim numeProdus As String
   Dim volumComanda As Integer
   Dim dataLivrare_1 As String
   Dim dataLivrare_2 As Integer
```

```
    For i = 1 To ZonaCitire_Comenzi.Rows.Count
        ZonaScriere_Comenzi.Cells(i, 1) = "C" + CStr(i)
        For j = 1 To ZonaCitire_Comenzi.Columns.Count
            If j = 1 Then
                numeProdus = ZonaCitire_Comenzi.Cells(i, j)

                ZonaScriere_Comenzi.Cells(i, j + 1) = numeProdus
            End If
            If j = 2 Then
                volumComanda = ZonaCitire_Comenzi.Cells(i, j)
                ZonaScriere_Comenzi.Cells(i, j + 1) = volumComanda
            End If
            If j = 3 Then
                dataLivrare_1 = ZonaCitire_Comenzi.Cells(i, j)
                ZonaScriere_Comenzi.Cells(i, j + 1) = 0
                ZonaScriere_Comenzi.Cells(i, j + 2) = 60 * DateDiff("n", "05/21/23", CDate(dataLivrare_1))
            End If
        Next j
    Next i
End Sub

Public Sub GenerareDateIntrare_Operatiuni()
    Zona_Operatiuni.Clear
    Dim contorOper As Long

    contorOper = 0

    Dim numeComanda As String
    Dim numeProdus As String
    Dim volum As Integer

    For i = 1 To ZonaScriere_Comenzi.Rows.Count
        numeComanda = ZonaScriere_Comenzi.Cells(i, 1)
        numeProdus = ZonaScriere_Comenzi.Cells(i, 2)
        volum = ZonaScriere_Comenzi.Cells(i, 3)

        For k = 1 To Zona_ProceseTeh.Rows.Count
            If numeProdus = Zona_ProceseTeh.Cells(k, 1) Then
                contorOper = contorOper + 1
                Zona_Operatiuni(contorOper, 1) = contorOper
                Zona_Operatiuni(contorOper, 2) = numeComanda
                Zona_Operatiuni(contorOper, 3) = Zona_ProceseTeh.Cells(k, 2)
                Zona_Operatiuni(contorOper, 4) = Zona_ProceseTeh.Cells(k, 3)
                Zona_Operatiuni(contorOper, 5) = volum * Zona_ProceseTeh.Cells(k, 4)
            End If
        Next k
    Next i
End Sub

Public Sub GenerareDateIntrare_ProcesTehnologic()
    Zona_ProceseTeh.Clear
    Zona_Precedente.Clear
```

```
    Dim numeProdus As String
    Dim numeUtilaj As String
    Dim norma As String
    Dim contorOperatii As Long
    Dim contorPrec As Long

    contorOperatii = 0

    For i = 1 To Zona_NormeTimp.Rows.Count
        numeProdus = Zona_Produse.Cells(i)

        For j = 1 To Zona_NormeTimp.Columns.Count
            If Len(Zona_NormeTimp.Cells(i, j)) > 0 Then
                contorOperatii = contorOperatii + 1
                numeUtilaj = "M" + CStr(j)
                norma = Zona_NormeTimp.Cells(i, j)

                Zona_ProceseTeh.Cells(contorOperatii, 1) = numeProdus
                Zona_ProceseTeh.Cells(contorOperatii, 2) = "Oper" + CStr(contorOperatii)
                Zona_ProceseTeh.Cells(contorOperatii, 3) = numeUtilaj
                Zona_ProceseTeh.Cells(contorOperatii, 4) = Round(60 * norma, 0)
            End If
        Next j
    Next i
End Sub

Public Sub GenerareLista_Precedente()
    Zona_Precedente.Clear

    Dim indOper_Old As Integer
    Dim indOper As Integer
    Dim numeProdus_Old As String
    Dim numeProdus As String
    Dim indScriere As Integer

    indOper_Old = Zona_Operatiuni.Cells(1, 1)
    numeProdus_Old = Zona_Operatiuni.Cells(1, 2)
    indScriere = 1

    For i = 2 To Zona_Operatiuni.Rows.Count
        indOper = Zona_Operatiuni.Cells(i, 1)
        numeProdus = Zona_Operatiuni.Cells(i, 2)

        If numeProdus = numeProdus_Old Then
            Zona_Precedente.Cells(indScriere, 1) = indOper
            Zona_Precedente.Cells(indScriere, 2) = indOper_Old

            indScriere = indScriere + 1
        End If

        indOper_Old = indOper
        numeProdus_Old = numeProdus
    Next i
```

```
End Sub
```

ANEXA 5. Script Visual Basic pentru aranjarea datelor în vederea folosirii lor de către modelul creat in CPLEX OPL CP

```
Dim Zona_DateDeCitit As Range
Dim Zona_DateDeVerificat As Range
Dim Zona_CodUtilaje As Range
Dim Zona_Centralizator As Range

Private Sub Worksheet_SelectionChange(ByVal Target As Range)
    Set Zona_DateDeCitit = Workbooks("Date_EF.xlsm").Names("Verificare_OPL_Intrari").RefersToRange
    Set Zona_DateDeVerificat = Workbooks("Date_EF.xlsm").Names("Verificare_OPL").RefersToRange

    Set Zona_CodUtilaje = Workbooks("Date_EF.xlsm").Names("VerificareCodUtilaj").RefersToRange
    Set Zona_Centralizator =
Workbooks("Date_EF.xlsm").Names("VerificareSolutie_CentralizatorUtilaje").RefersToRange

    Dim ZonaClick As Range
    Set ZonaClick = Range("A1")

    If Not Application.Intersect(ZonaClick, Range(Target.Address)) Is Nothing Then
        Call SortareDateOPL
        Call VerificareSolutieOPL
        MsgBox ("gata")
    End If
End Sub

Public Sub SortareDateOPL()
    Zona_DateDeVerificat.ClearContents
    Zona_DateDeCitit.Copy
    Zona_DateDeVerificat.PasteSpecial
    Zona_CodUtilaje.ClearContents

    Dim LastRow As Long
    LastRow = Range("H" & Rows.Count).End(xlUp).Row
    Dim numeUtilaj As String

    For i = 2 To Zona_DateDeVerificat.Rows.Count
        numeUtilaj = Zona_DateDeVerificat.Cells(i, 5)
        Zona_CodUtilaje.Cells(i - 1, 1) = Mid(numeUtilaj, 2)
    Next i

    Range("D2:I" & LastRow).Sort key1:=Range("I2"), order1:=xlAscending, key2:=Range("F2"),
order2:=xlAscending, Header:=xlYes
End Sub
Public Sub VerificareSolutieOPL()
```

```
    Zona_Centralizator.ClearContents

    Dim utilajC As String
    Dim utilajA As String

    Dim timpInceput As Long
    Dim timpSfarsit As Long
    Dim timpSfarsitAnterior As Long
    Dim indiceScriere As Integer
    indiceScriere = 1

    utilajC = Zona_DateDeVerificat.Cells(2, 5)
    timpInceput = Zona_DateDeVerificat.Cells(2, 3)
    timpSfarsit = Zona_DateDeVerificat.Cells(2, 4)

    For i = 3 To Zona_DateDeVerificat.Rows.Count
        If utilajC <> Zona_DateDeVerificat.Cells(i, 5) Then
            Zona_Centralizator.Cells(indiceScriere, 1) = utilajC
            Zona_Centralizator.Cells(indiceScriere, 2) = 1
            Zona_Centralizator.Cells(indiceScriere, 3) = timpInceput
            Zona_Centralizator.Cells(indiceScriere, 4) = timpSfarsit

            indiceScriere = indiceScriere + 1
            timpInceput = Zona_DateDeVerificat.Cells(i, 3)
        Else
            If timpSfarsit > Zona_DateDeVerificat.Cells(i, 3) Then
                MsgBox ("Solutie gresita")

                Zona_Centralizator.Cells(indiceScriere, 1) = utilajC
                Zona_Centralizator.Cells(indiceScriere, 2) = 0
                Zona_Centralizator.Cells(indiceScriere, 3) = timpInceput
                Zona_Centralizator.Cells(indiceScriere, 4) = timpSfarsit

                indiceScriere = indiceScriere + 1

                Do
                    utilajC = Zona_DateDeVerificat.Cells(i, 5)
                    i = i + 1
                Loop While utilajC = Zona_DateDeVerificat.Cells(i, 5)

                utilajC = Zona_DateDeVerificat.Cells(i, 5)
                timpInceput = Zona_DateDeVerificat.Cells(i, 3)
                timpSfarsit = Zona_DateDeVerificat.Cells(i, 4)

            End If
        End If

        utilajC = Zona_DateDeVerificat.Cells(i, 5)
        timpSfarsit = Zona_DateDeVerificat.Cells(i, 4)
    Next i
End Sub
```

ANEXA 6. Codul Algoritmului Genetic folosit în Matlab.

crossover

```
function offspring = crossover(selectedParents)
    [numParentsr,numParentsc, numGenes] = size(selectedParents);
    offspring = zeros(size(selectedParents));


        crossoverPoint = randi([1, numGenes-1]); % Random crossover point

        offspring(1:crossoverPoint,:, 1) = selectedParents(1:crossoverPoint,:, 1);
        offspring(crossoverPoint+1:end,:, 1) = selectedParents(crossoverPoint+1:end,:,
2);

        offspring(1:crossoverPoint,:, 2) = selectedParents(1:crossoverPoint,:, 2);
        offspring(crossoverPoint+1:end,:, 2) = selectedParents(crossoverPoint+1:end,:,
1);

end
```

initializePopulation

```
function [population] = initializePopulation(popSize, numOrders, numHours, prod_cen,
ordersData, ppData)
population = false(numOrders, numHours, popSize); % Initialize with zeros (false)



numProducts = height(ppData);
numMachines = 57; % Total machines available

% Initialize the machine matrix with false
```

```
machineMatrix = false(numProducts, numMachines);

for k=1:popSize
    % Iterate over each product
    for i = 1:numProducts
        % Iterate over each machine starting from the 3rd column in ppData
        for j = 1:numMachines
            value = ppData{i, j+2};
            if ~isnan(value) && value ~= 0 && ~isempty(value)
                machineMatrix(i, j) = true;
            else
                machineMatrix(i, j) = false;
            end
        end
    end
    orderStart = ordersData{:, 'Var10'}; % Use actual column name
    orderEnd = ordersData{:, 'Var11'}; % Use actual column name
    prod_num = ordersData{:, 'Var4'};
    for i=1:length(prod_num)
        productionTime(i,1) = (ppData{prod_num(i), 'TimpTotalDeProd'} * ordersData{i,
'Var5'})/60; % Total hours needed
    end
    latestStart = orderEnd - ceil(productionTime);
    av_win=latestStart-orderStart;

    for orderIdx = 1:numOrders
        orderIdx
        if orderIdx ==1
            startHour = randi([orderStart(1), latestStart(1)]);
            population(orderIdx, startHour:startHour+ceil(productionTime(1))-1,
k)=true;
            continue;
        end
        % Find a random start time within the valid window
        if orderStart(orderIdx) < latestStart(orderIdx)
            %      startHour = randi([orderStart(orderIdx) , latestStart(orderIdx)]);
            truecol=find(machineMatrix(prod_num(orderIdx),:));

            updc=machinechk(orderIdx, machineMatrix, population, truecol, prod_num, k);
            if ~isempty(updc)
                % Ensure the array is sorted
                numbers = sort(updc);

                % Initialize variables
                lowerLimits = numbers(1); % Start with the first number as the initial
lower limit
                upperLimits = [];
                prevNumber = numbers(1);

                % Iterate through the array starting from the second element
                for i = 2:length(numbers)
                    if numbers(i) == prevNumber + 1
                        % If the current number is consecutive, move to the next
                        prevNumber = numbers(i);
                    else
                        % If not consecutive, end the current window and start a new
one
```

```
                    upperLimits(end+1) = prevNumber; % End the current window
                    lowerLimits(end+1) = numbers(i); % Start a new window
                    prevNumber = numbers(i);
                end
            end
            % After the loop, add the last upper limit
            upperLimits(end+1) = numbers(end);


            for j=1:length(upperLimits)
                ord_lim=[orderStart(orderIdx) , latestStart(orderIdx)];
                ov11=max(ord_lim(1),lowerLimits(j));
                ov12=min(ord_lim(2),upperLimits(j));
                updc=[lowerLimits(j),upperLimits(j)];
                if ov11<ov12
                    if (lowerLimits(j)-ord_lim(1))>= (ord_lim(2)-upperLimits(j))
                        ord_lim1=[ord_lim(1) , lowerLimits(j)];
                        ord_lim=ord_lim1;
                        if j==length(upperLimits)
                            if ord_lim(1)>ord_lim(2)
                                ord_lim(1)=ord_lim(2);
                            end
                            startHour = randi([ord_lim(1) , ord_lim(2)]);
                            break;
                        end
                    elseif (lowerLimits(j)-ord_lim(1))< (ord_lim(2)-upperLimits(j))
                        ord_lim1=[upperLimits(j) , ord_lim(2)];
                        ord_lim=ord_lim1;
                        if j==length(upperLimits)
                            if ord_lim(1)>ord_lim(2)
                                ord_lim(1)=ord_lim(2);
                            end
                            startHour = randi([ord_lim(1) , ord_lim(2)]);
                            break;
                        end
                    end
                end

            end


        else
            startHour = orderStart;
        end

        % Set the hours for this order as busy
        population(orderIdx, startHour:(startHour+(ceil(productionTime(orderIdx))-
1)),k) = true;
    end
end
end
```

machinecheck

```
function [notedColumns] = machinechk(orderIdx,machineMatrix, population, truecol,
prod_num, k)
```

```
% Initialize an array to hold rows from previous orders with the same true column.
previousOrderRows = [];

% Loop through all previous orders
for i = 1:(orderIdx-1)
    % Check if any of the trueCols for the current order are also true in the previous
orders
    if any(machineMatrix(prod_num(i), truecol))
        % If so, add this row number to the list
        previousOrderRows = [previousOrderRows, i];
    end
end


notedColumns = [];
updc=[];
for i = previousOrderRows
    % Find columns that are true for the particular rows in populationMatrix
    notedColsCurrentRow = find(population(i, :, k));
    % Combine these columns for all identified rows
    notedColumns = union(notedColumns, notedColsCurrentRow);
end
updc=[min(notedColumns),max(notedColumns)];
end
```

mutation

```
function mutatedOffspring = mutation(offspring)
    [numRows, numCols, numSlices] = size(offspring);
    mutatedOffspring = offspring; % Copy the offspring
    mutationRate = 0.7; % Probability of mutating a given row

    % Iterate over each slice
    for z = 1:numSlices
        % Iterate through each row
        for row = 1:numRows
            if rand() < mutationRate
                % Perform a random test for the current row
                randomTest = randi([-2, 2], 1, 1);

                % Find indices of consecutive ones in the row
                rowDiff = diff([0, offspring(row, :, z), 0]);
                startIndices = find(rowDiff == 1);
                endIndices = find(rowDiff == -1) - 1;

                % Clear the current row to prepare for mutation
                mutatedOffspring(row, :, z) = false;

                % Loop through each group of consecutive ones
                for i = 1:length(startIndices)
                    if randomTest == 1 % Shift right
                        newStart = min(numCols, startIndices(i) + 1);
                        newEnd = min(numCols, endIndices(i) + 1);
                        mutatedOffspring(row, newStart:newEnd, z) = true;
                    elseif randomTest == -1 % Shift left
                        newStart = max(1, startIndices(i) - 1);
                        newEnd = max(1, endIndices(i) - 1);
```

```
                    mutatedOffspring(row, newStart:newEnd, z) = true;
                else

                    mutatedOffspring(row, startIndices(i):endIndices(i), z) = true;
                end
            end
        end
    end
end
```

operation

```
clc
clear

% Load Orders Data
orders = readtable('Orders1.2.xlsx', 'Sheet', 'Table1');
% Load Production Planning Data
ppData = readtable('PP dec 2023.xlsx', 'Sheet', 'tmpb');


% GA Parameters
popSize = 6; % Example population size
maxGenerations = 30; % Example number of generations
numOrders = 280;
numHours = 8760;
prod_cen = 1;
gbestFitness=inf;
gbestsol=[];
%%%%%%%%%%%%%%%%%%%%  Initialization Stage %%%%%%%%%%%%%%%%%%%%%%
% Initialize Population
population = initializePopulation(popSize, numOrders, numHours, prod_cen,orders,
ppData);

for j=1:popSize
% Calculate total production time for the first individual in the population
    totalProductionTime(j,1) = sum(any(population(:,:,j), 1));
end

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%load ('op.mat');


%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%  GA loop  %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
for gen = 1:maxGenerations
    % Selection
    selectedParents = selection(population, totalProductionTime);

    % Crossover
    offspring = crossover(selectedParents);

    % Mutation
    mutatedOffspring = mutation(offspring);

    up_pop=cat(3, selectedParents, offspring, mutatedOffspring);
    population=up_pop;
    for j=1:popSize
```

```
    % Calculate total production time for the first individual in the population
        totalProductionTime(j,1) = sum(any(population(:,:,j), 1));
    end

    % Select the best individual as the potential solution
    [bestFitness, bestIndex] = min(totalProductionTime);
    bestSolution = population(:, :, bestIndex);
    if bestFitness<gbestFitness
        gbestFitness=bestFitness;
        gbestsol=bestSolution;
    end
    gen
end

disp([' Best Fitness = ', num2str(gbestFitness)]);
disp('Fisierul excell a fost generat');

%%

gbestsol_reshaped = reshape(gbestsol, numOrders, 24, 365);

B_temp = any(gbestsol_reshaped, 2);

B = squeeze(B_temp);
gbestsol1=B;
%%
%jobnum = orders{:, 'Var16'}; % Use actual column name
jobnum = [0:numOrders]';
newRow = 1:365;
gbestsol1 = [newRow; gbestsol1];
%newColumn = [0;jobnum];
gbestsol1 = [jobnum, gbestsol1];

filename = 'gbestsol.xlsx';

xlswrite(filename, gbestsol1);

%%

% figure;
% imagesc(gbestsol1);
% colormap(gray); %
% xlabel('Days');
% ylabel('Task ID');
% title('gbestsol Visualization');


%%

[nTasks, nTimeSlots] = size(gbestsol1);
figure; hold on;
for i = 1:nTasks
    for j = 1:nTimeSlots
        if gbestsol1(i, j)
            plot([j, j], [i-0.5, i+0.5], 'b', 'LineWidth', 2); %
        end
    end
```

```
end
xlabel('Days');
ylabel('Task ID');
title('Schedule Visualization in Days');
axis tight;


%%
[nTasks, nTimeSlots] = size(gbestsol);
figure; hold on;
for i = 1:nTasks
    for j = 1:nTimeSlots
        if gbestsol(i, j)
            plot([j, j], [i-0.5, i+0.5], 'b', 'LineWidth', 2); %
        end
    end
end
xlabel('Hours');
ylabel('Task ID');
title('Schedule Visualization in hours');
axis tight;
```

selection

```
function selectedParents = selection(population, fitnessScores)
    numParents=2;
    tournamentSize=2;
    numIndividuals = size(population, 3);
    selectedParents = zeros(280, 8760); % Initialize matrix for selected parents


    for i = 1:numParents
        % Randomly select 'tournamentSize' individuals for the tournament
        indices = randperm(numIndividuals, tournamentSize);
        tournamentIndividuals = population(:, :, indices);
        tournamentFitness = fitnessScores(indices);

        % Find the best individual in the tournament
        [~, bestIndex] = min(tournamentFitness);
        bestIndividual = population(:, :, bestIndex);

        % Add the best individual to the selected parents
        selectedParents(:, : ,i) = bestIndividual;
    end
end
```

ANEXA 7. Explicații ale modului în care funcționeaza Algoritmul Genetic

Constrain No 1 (machinechk.m, lines 4 to 24)

Correspondence between Constraint and Variables:

- Constraint: **T_in,mp - T_jn,mp + KZ_in,jn,mp >= d_jn,mp**

  - This constraint ensures that the processing times do not overlap on the same machine for different jobs.

Variables in Algorithm:

- **orderIdx**: Index of the current order.
- **machineMatrix**: Matrix indicating the availability of machines for each product.
- **population**: 3D matrix representing the current population.
- **truecol**: Columns corresponding to the machines available for the current order's product.
- **prod_num:** Array of product numbers for each order.
- **k:** Index of the current individual in the population.

Specific Lines and Correspondence:

1. Line 4:
   - **previousOrderRows** = [];
   - Initializes an array to hold rows from previous orders with the same true column (machine).
2. Lines 5-10:
   - Loop through all previous orders to check if any of the **trueCols** for the current order are also true in the previous orders.
   - if any**(machineMatrix(prod_num(i), truecol))**
     - This line checks if the same machine is being used for any previous orders.
   - **previousOrderRows = [previousOrderRows, i];**
     - Adds the row number of the previous order to **previousOrderRows** if the same machine is used.

3. Lines 11-24:
   - Identify active machine slots for the identified previous orders.
   - **notedColumns = [];**
     - Initializes an array to hold the active slots.
   - Loop through **previousOrderRows:**

- **notedColsCurrentRow = find(population(i, :, k));**
  - Finds columns (time slots) that are true (active) for the identified rows.
- **notedColumns = union(notedColumns, notedColsCurrentRow);**
  - Combines these columns to determine the active slots for all previous orders.
- **updc = [min(notedColumns), max(notedColumns)];**
  - Sets the minimum and maximum of the noted columns to ensure that the current order does not overlap with these slots.

Constrain No 2 (machinechk.m, lines 4 to 24)

Correspondence between Constraint and Variables:

- Constraint: **T_jn,mp - T_in,mp + K(1 - Z_in,jn,mp) >= d_in,mp**
  - This constraint ensures that if job jn starts after job in, the processing time of job in is respected, preventing any overlap.

Variables in Algorithm:

- **orderIdx:** Index of the current order.
- **machineMatrix:** Matrix indicating the availability of machines for each product.
- **population:** 3D matrix representing the current population.
- **truecol:** Columns corresponding to the machines available for the current order's product.
- **prod_num:** Array of product numbers for each order.
- **k:** Index of the current individual in the population.

Specific Lines and Correspondence:

1. Line 4:
   - **previousOrderRows = [];**
   - Initializes an array to hold rows from previous orders with the same true column (machine).
2. Lines 5-10:
   - Loop through all previous orders to check if any of the **trueCols** for the current order are also true in the previous orders.
   - **if any(machineMatrix(prod_num(i), truecol))**
     - This line checks if the same machine is being used for any previous orders.
   - **previousOrderRows = [previousOrderRows, i];**
     - Adds the row number of the previous order to **previousOrderRows** if the same machine is used.
3. Lines 11-24:
   - Identify active machine slots for the identified previous orders.

- **notedColumns = [];**

  - Initializes an array to hold the active slots.

- Loop through previousOrderRows:

  - **notedColsCurrentRow = find(population(i, :, k));**

    - Finds columns (time slots) that are true (active) for the identified rows.

  - **notedColumns = union(notedColumns, notedColsCurrentRow);**

    - Combines these columns to determine the active slots for all previous orders.

- **updc = [min(notedColumns), max(notedColumns)];**

  - Sets the minimum and maximum of the noted columns to ensure that the current order does not overlap with these slots.

Constrain No 3 (machinechk.m, lines 4 to 24)

Correspondence between Constraint and Variables:

- Constraint: **T_in,mp+1 - T_in,mp >= d_in,mp**

  - This constraint ensures that the processing of job in on machine mp+1 starts only after the job is finished on machine mp, avoiding any overlap of the same job across machines and ensuring a logical and sequential workflow.

Variables in Algorithm:

- **orderIdx:** Index of the current order.
- **machineMatrix:** Matrix indicating the availability of machines for each product.
- **population**: 3D matrix representing the current population.
- **truecol:** Columns corresponding to the machines available for the current order's product.
- **prod_num:** Array of product numbers for each order.
- **k:** Index of the current individual in the population.

Specific Lines and Correspondence:

1. Line 4:

   - **previousOrderRows = [];**

   - Initializes an array to hold rows from previous orders with the same true column (machine).

2. Lines 5-10:

   - Loop through all previous orders to check if any of the **trueCols** for the current order are also true in the previous orders.

   - **if any(machineMatrix(prod_num(i), truecol))**

     - This line checks if the same machine is being used for any previous orders.

- previousOrderRows = [previousOrderRows, i];

- Adds the row number of the previous order to previousOrderRows if the same machine is used.

3. Lines 11-24:

- Identify active machine slots for the identified previous orders.

- notedColumns = [];

- Initializes an array to hold the active slots.

- Loop through previousOrderRows:

- notedColsCurrentRow = find(population(i, :, k));

- Finds columns (time slots) that are true (active) for the identified rows.

- notedColumns = union(notedColumns, notedColsCurrentRow);

- Combines these columns to determine the active slots for all previous orders.

- updc = [min(notedColumns), max(notedColumns)];

- Sets the minimum and maximum of the noted columns to ensure that the current order does not overlap with these slots.

Constrain No 4 (initializePopulation.m, lines 43 to 108)

Correspondence between Constraint and Variables:

- Constraint: **T_in,1 >= d_int_i**

- This constraint ensures that the start time T_in,1 for job in is not earlier than the earliest possible start time d_int_i.

Variables in Algorithm:

- ordersData: Data of the orders.

- ppData: Production planning data.

- startHour: The hour when the order starts.

- **orderStart**: Array of start times for orders.

- latestStart: Array of the latest start times for orders.

- productionTime: Total hours needed for production of each order.

Specific Lines and Correspondence:

1. Lines 43-46:

- Extract the start times **(orderStart)** and end times **(orderEnd)** from **ordersData.**

- Calculate the **latestStart** times for each order.

- Calculate **productionTime** for each order.

- This ensures the boundary conditions for the start times are respected.

2. Lines 47-50:

- For each order, find a random **startHour** within the available window defined by **orderStart** and **latestStart**.

- This ensures that the start time **T_in,1** for job in is not earlier than the earliest possible start time **d_int_i.**

3. Lines 51-108:

- These lines ensure that the generated schedule respects the boundary conditions set during initialization.

- Specifically, they ensure that the start times are within the allowed range, and no order starts before its earliest possible start time.

Implementation in Lines 43-108:

- These lines ensure that the start times for each job are within the allowed range, respecting the constraint **T_in,1 >= d_int_i.**

Constrain No 5 (initializePopulation.m, lines 43 to 108)

Correspondence between Constraint and Variables:

- Constraint: **T_in,mp <= d_li - d_in,mp**

- This constraint ensures that the start time **T_in,mp** for job in on machine mp is not later than the latest possible start time **d_li - d_in,mp**, ensuring that no job starts after it is supposed to.

Variables in Algorithm:

- ordersData: Data of the orders.

- ppData: Production planning data.

- startHour: The hour when the order starts.

- orderEnd: Array of end times for orders.

- latestStart: Array of the latest start times for orders.

- **productionTime**: Total hours needed for production of each order.

Specific Lines and Correspondence:

1. Lines 43-46:

- Extract the start times **(orderStart)** and end times **(orderEnd)** from **ordersData.**

- Calculate the **latestStart** times for each order.

- Calculate **productionTime** for each order.

- This ensures the boundary conditions for the start times are respected, particularly ensuring jobs do not start after their latest start time.

2. Lines 47-50:

- For each order, find a random **startHour** within the available window defined by **orderStart** and **latestStart.**

- This ensures that the start time **T_in,mp** for job in is within the allowed window and does not exceed the latest possible start time.

3. Lines 51-108:

- These lines ensure that the generated schedule respects the boundary conditions set during initialization.

- Specifically, they ensure that the start times are within the allowed range, and no order starts after its latest possible start time.

Implementation in Lines 43-108:

- These lines ensure that the start times for each job are within the allowed range, respecting the constraint **T_in,mp <= d_li - d_in,mp.**

Condition 1&2 (initializePopulation.m, lines 43 to 108)

Correspondence between Condition and Variables:

- Condition: The production time for each order when received is 6 weeks (6 * 5 * 16 hrs = 480 hrs).

- This condition sets the production window for each order to 480 hours.

Variables in Algorithm:

- ordersData: Data of the orders.

- ppData: Production planning data.

- productionTime: Total hours needed for production of each order.

- startHour: The hour when the order starts.

- orderStart: Array of start times for orders.

- orderEnd: Array of end times for orders.

- latestStart: Array of the latest start times for orders.

Specific Lines and Correspondence:

1. Lines 43-46:

- Extract the start times (**orderStart**) and end times (**orderEnd**) from **ordersData**.

- Calculate **productionTime** for each order:

- productionTime(i,1) = (ppData{prod_num(i), 'TimpTotalDeProd'} * ordersData{i, 'Var2'}) / 60;

- This ensures that the production time is set correctly based on the data, including the 480-hour constraint.

2. Lines 47-50:

- For each order, find a random **startHour** within the available window defined by **orderStart** and **latestStart.**

- This ensures that the start times respect the production window of 480 hours.

3. Lines 51-108:

- These lines ensure that the generated schedule respects the boundary conditions set during initialization.

- Specifically, they ensure that the start times and production times are within the allowed range, including the 480-hour constraint.

ANEXA 8. Codul Algoritmului Hybrid (AG+PSO)

crossover

```
% Crossover function with percentage-based crossover point
function offspring = crossover(parents, crossoverPercentage)
    [numGenes, numFeatures, numParents] = size(parents);
    numOffspring = numParents / 2;
    offspring = zeros(numGenes, numFeatures, numOffspring);

    for i = 1:numOffspring
        parent1 = parents(:, :, 2*i-1);
        parent2 = parents(:, :, 2*i);

        % Calculate crossover point based on the given percentage
        crossoverPoint = ceil((crossoverPercentage / 100) * numGenes);

        % Concatenate genes from two parents at the crossover point
        % Ensuring both parts have compatible dimensions
        for f = 1:numFeatures
            offspring(1:crossoverPoint, f, i) = parent1(1:crossoverPoint, f);
            offspring(crossoverPoint+1:end, f, i) = parent2(crossoverPoint+1:end, f);
        end
    end
end
```

initializePopulation

```
function [population] = initializePopulation(popSize, numOrders, numHours, prod_cen,
ordersData, ppData)
population = false(numOrders, numHours, popSize); % Initialize with zeros (false)



numProducts = height(ppData);
numMachines = 57; % Total machines available

% Initialize the machine matrix with false
machineMatrix = false(numProducts, numMachines);

for k=1:popSize
    % Iterate over each product
    for i = 1:numProducts
        % Iterate over each machine starting from the 3rd column in ppData
        for j = 1:numMachines
            value = ppData{i, j+2};
            if ~isnan(value) && value ~= 0 && ~isempty(value)
                machineMatrix(i, j) = true;
            else
                machineMatrix(i, j) = false;
            end
        end
    end
    orderStart = ordersData{:, 'Var10'}; % Use actual column name
    orderEnd = ordersData{:, 'Var11'}; % Use actual column name
    prod_num = ordersData{:, 'Var4'};
    for i=1:length(prod_num)
        productionTime(i,1) = (ppData{prod_num(i), 'TimpTotalDeProd'} * ordersData{i,
'Var5'})/60; % Total hours needed
    end
    latestStart = orderEnd - ceil(productionTime);
    av_win=latestStart-orderStart;

    for orderIdx = 1:numOrders
        orderIdx
        if orderIdx ==1
            startHour = randi([orderStart(1), latestStart(1)]);
            population(orderIdx, startHour:startHour+ceil(productionTime(1))-1,
k)=true;
            continue;
        end
        % Find a random start time within the valid window
        if orderStart(orderIdx) < latestStart(orderIdx)
            %      startHour = randi([orderStart(orderIdx) , latestStart(orderIdx)]);
            truecol=find(machineMatrix(prod_num(orderIdx),:));

            updc=machinechk(orderIdx, machineMatrix, population, truecol, prod_num, k);
            if ~isempty(updc)
                % Ensure the array is sorted
                numbers = sort(updc);

                % Initialize variables
                lowerLimits = numbers(1); % Start with the first number as the initial
lower limit
                upperLimits = [];
```

```
                prevNumber = numbers(1);

                % Iterate through the array starting from the second element
                for i = 2:length(numbers)
                    if numbers(i) == prevNumber + 1
                        % If the current number is consecutive, move to the next
                        prevNumber = numbers(i);
                    else
                        % If not consecutive, end the current window and start a new
one
                        upperLimits(end+1) = prevNumber; % End the current window
                        lowerLimits(end+1) = numbers(i); % Start a new window
                        prevNumber = numbers(i);
                    end
                end
                % After the loop, add the last upper limit
                upperLimits(end+1) = numbers(end);


                for j=1:length(upperLimits)
                    ord_lim=[orderStart(orderIdx) , latestStart(orderIdx)];
                    ov11=max(ord_lim(1),lowerLimits(j));
                    ov12=min(ord_lim(2),upperLimits(j));
                    updc=[lowerLimits(j),upperLimits(j)];
                    if ov11<ov12
                        if (lowerLimits(j)-ord_lim(1))>= (ord_lim(2)-upperLimits(j))
                            ord_lim1=[ord_lim(1) , lowerLimits(j)];
                            ord_lim=ord_lim1;
                            if j==length(upperLimits)
                                if ord_lim(1)>ord_lim(2)
                                    ord_lim(1)=ord_lim(2);
                                end
                                startHour = randi([ord_lim(1) , ord_lim(2)]);
                                break;
                            end
                        elseif (lowerLimits(j)-ord_lim(1))< (ord_lim(2)-upperLimits(j))
                            ord_lim1=[upperLimits(j) , ord_lim(2)];
                            ord_lim=ord_lim1;
                            if j==length(upperLimits)
                                if ord_lim(1)>ord_lim(2)
                                    ord_lim(1)=ord_lim(2);
                                end
                                startHour = randi([ord_lim(1) , ord_lim(2)]);
                                break;
                            end
                        end
                    end
                end

            end


        else
            startHour = orderStart;
        end

        % Set the hours for this order as busy
```

```
        population(orderIdx, startHour:(startHour+(ceil(productionTime(orderIdx))-
1)),k) = true;
    end
end
end
```

machinecheck

```
function [notedColumns] = machinechk(orderIdx,machineMatrix, population, truecol,
prod_num, k)

% Initialize an array to hold rows from previous orders with the same true column.
previousOrderRows = [];

% Loop through all previous orders
for i = 1:(orderIdx-1)
    % Check if any of the trueCols for the current order are also true in the previous
orders
    if any(machineMatrix(prod_num(i), truecol))
        % If so, add this row number to the list
        previousOrderRows = [previousOrderRows, i];
    end
end


notedColumns = [];
updc=[];
for i = previousOrderRows
    % Find columns that are true for the particular rows in populationMatrix
    notedColsCurrentRow = find(population(i, :, k));
    % Combine these columns for all identified rows
    notedColumns = union(notedColumns, notedColsCurrentRow);
end
updc=[min(notedColumns),max(notedColumns)];
end
```

mutation

```
function mutatedOffspring = mutation(offspring)
    [numRows, numCols, numSlices] = size(offspring);
    mutatedOffspring = offspring; % Copy the offspring
    mutationRate = 0.6; % Probability of mutating a given row

    % Iterate over each slice
    for z = 1:numSlices
        % Iterate through each row
        for row = 1:numRows
            if rand() < mutationRate
                % Perform a random test for the current row
                randomTest = randi([-2, 2], 1, 1);

                % Find indices of consecutive ones in the row
                rowDiff = diff([0, offspring(row, :, z), 0]);
                startIndices = find(rowDiff == 1);
                endIndices = find(rowDiff == -1) - 1;

                % Clear the current row to prepare for mutation
```

```
                mutatedOffspring(row, :, z) = false;

                % Loop through each group of consecutive ones
                for i = 1:length(startIndices)
                    if randomTest == 1 % Shift right
                        newStart = min(numCols, startIndices(i) + 1);
                        newEnd = min(numCols, endIndices(i) + 1);
                        mutatedOffspring(row, newStart:newEnd, z) = true;
                    elseif randomTest == -1 % Shift left
                        newStart = max(1, startIndices(i) - 1);
                        newEnd = max(1, endIndices(i) - 1);
                        mutatedOffspring(row, newStart:newEnd, z) = true;
                    else

                        mutatedOffspring(row, startIndices(i):endIndices(i), z) = true;
                    end
                end
            end
        end
    end
end
```

operation

```
clc
clear

% Load Orders Data
orders = readtable('Orders1.2.xlsx', 'Sheet', 'Table1');
% Load Production Planning Data
ppData = readtable('PP dec 2023.xlsx', 'Sheet', 'tmpb');

% Parameters
popSize = 6; % Example population size
numParticles = 30; % Number of particles for PSO
maxGenerations = 50; % Example number of generations
numOrders = 280;
numHours = 8760;
prod_cen = 1;
gbestFitness = inf;
gbestsol = [];
w = 0.5; % Inertia weight
c1 = 1.5; % Cognitive (personal best) weight
c2 = 1.5; % Social (global best) weight
crossoverPercentage = 90; % Example percentage for crossover
tournamentSize = 5;
numParents = 20;

% Initialize Population and PSO variables
population = initializePopulation(popSize, numOrders, numHours, prod_cen, orders,
ppData);
velocity = zeros(size(population)); % Initialize velocity
pBest = population; % Personal best
pBestFitness = inf(popSize, 1); % Personal best fitness
gBest = population(:, :, 1); % Global best

% Calculate initial production time for the population
```

```
for j = 1:popSize
    totalProductionTime(j, 1) = sum(any(population(:, :, j), 1));
    if totalProductionTime(j, 1) < pBestFitness(j)
        pBest(:, :, j) = population(:, :, j);
        pBestFitness(j) = totalProductionTime(j, 1);
    end
    if totalProductionTime(j, 1) < gbestFitness
        gbestFitness = totalProductionTime(j, 1);
        gBest = population(:, :, j);
        gbestsol = gBest; % Update global best solution
    end
end

% Main loop
for gen = 1:maxGenerations
    gen
    % PSO Step: Update velocity and position
    for i = 1:popSize
        r1 = rand(size(population(:, :, i)));
        r2 = rand(size(population(:, :, i)));
        velocity(:, :, i) = w * velocity(:, :, i) ...
                          + c1 * r1 .* (pBest(:, :, i) - population(:, :, i)) ...
                          + c2 * r2 .* (gBest - population(:, :, i));
        population(:, :, i) = population(:, :, i) + velocity(:, :, i);

        % Ensure positions are within valid bounds (optional, if applicable)
        % population(:, :, i) = max(min(population(:, :, i), upperBound), lowerBound);
    end

    % Evaluate fitness of the updated population
    for j = 1:popSize
        totalProductionTime(j, 1) = sum(any(population(:, :, j), 1));
        if totalProductionTime(j, 1) < pBestFitness(j)
            pBest(:, :, j) = population(:, :, j);
            pBestFitness(j) = totalProductionTime(j, 1);
        end
        if totalProductionTime(j, 1) < gbestFitness
            gbestFitness = totalProductionTime(j, 1);
            gBest = population(:, :, j);
            gbestsol = gBest; % Update global best solution
        end
    end

    % GA Step: Selection, Crossover, Mutation
    selectedParents = selection(population, totalProductionTime, numParents,
tournamentSize, gBest);
    offspring = crossover(selectedParents, crossoverPercentage);
    mutatedOffspring = mutation(offspring);

    % Combine populations (PSO updated and GA offspring)
    population = cat(3, population, mutatedOffspring);
    totalProductionTime = [totalProductionTime; inf(size(mutatedOffspring, 3), 1)]; %
Extend the totalProductionTime array

    % Select the best individuals
    [population, totalProductionTime] = selectBest(population, totalProductionTime,
popSize); % Select the best individuals
```

```
    % Recalculate total production time for the combined population
    for j = 1:popSize
        totalProductionTime(j, 1) = sum(any(population(:, :, j), 1));
    end

    % Select the best individual as the potential solution
    [bestFitness, bestIndex] = min(totalProductionTime);
    bestSolution = population(:, :, bestIndex);
    if bestFitness < gbestFitness
        gbestFitness = bestFitness;
        gbestsol = bestSolution;
    end
end

disp([' Best Fitness = ', num2str(gbestFitness)]);
disp('Fisierul excell a fost generat');

% Post-process the best solution for output
if isempty(gbestsol)
    error('gbestsol is empty. The algorithm did not find a valid solution.');
end

numTasks = size(gbestsol, 1); % Determine the number of tasks from gbestsol
numTimeSlots = size(gbestsol, 2); % Determine the number of time slots from gbestsol
gbestsol_reshaped = reshape(gbestsol, numTasks, 24, numTimeSlots / 24); % Adjusted
reshape dimensions

B_temp = any(gbestsol_reshaped, 2);
B = squeeze(B_temp);
gbestsol1 = B;

%% Prepare the solution for Excel output
%jobnum = orders{:, 'Var1'}; % Use actual column name
jobnum = [0:numOrders]';
newRow = (1:size(gbestsol1, 2)); % Adjust the newRow to match gbestsol1's size
gbestsol1 = [newRow; gbestsol1];
%newColumn = ['0'; jobnum];
%newColumn = newColumn(1:size(gbestsol1, 1)); % Adjust the newColumn size to match
gbestsol1
%newColumn=cell2mat(newColumn);
gbestsol1 = [jobnum, gbestsol1];

filename = 'gbestsol.xlsx';
xlswrite(filename, gbestsol1);

% Visualization of the final schedule
[nTasks, nTimeSlots] = size(gbestsol1);
figure; hold on;
for i = 1:nTasks
    for j = 1:nTimeSlots
        if gbestsol1(i, j)
            plot([j, j], [i-0.5, i+0.5], 'b', 'LineWidth', 2);
        end
    end
end
xlabel('Zile');
```

```
ylabel('Numarul Comenzii');
title('Diagrama Gantt exprimata in numar zile');
axis tight;

[nTasks, nTimeSlots] = size(gbestsol);
figure; hold on;
for i = 1:nTasks
    for j = 1:nTimeSlots
        if gbestsol(i, j)
            plot([j, j], [i-0.5, i+0.5], 'b', 'LineWidth', 2);
        end
    end
end
xlabel('Ore');
ylabel('Numarul Comenzii');
title('Diagrama Gantt exprimata in ore');
axis tight;
```

selection

```
function selectedParents = selection(population, fitnessScores)
    numParents=2;
    tournamentSize=2;
    numIndividuals = size(population, 3);
    selectedParents = zeros(280, 8760); % Initialize matrix for selected parents


    for i = 1:numParents
        % Randomly select 'tournamentSize' individuals for the tournament
        indices = randperm(numIndividuals, tournamentSize);
        tournamentIndividuals = population(:, :, indices);
        tournamentFitness = fitnessScores(indices);

        % Find the best individual in the tournament
        [~, bestIndex] = min(tournamentFitness);
        bestIndividual = population(:, :, bestIndex);

        % Add the best individual to the selected parents
        selectedParents(:, : ,i) = bestIndividual;
    end
end
```

L I S T A

Lucrărilor ştiinţifice în domeniul Inginerie și Management

A1. LUCRĂRI INDEXATE ZENODO:

Lungu F., Teodorescu M., ”Internet of things (IoT) și procesele logistice industriale” în cadrul conferinței ”Management Challenges Within Globalization, RMEE, 17-19 septembrie 2020 ”, Universitatea Tehnică din Cluj Napoca, Todesco Publishing House, ISSN 2247 – 8639, ISSN-L = 2247 – 8639, pp 650-659, indexata ZENODO, Disponibil la: _https://zenodo.org/record/6557515#.Yx8Nc3ZBzdU

A2. LUCRĂRI INDEXATE BDI :

Lungu F., Teodorescu M., Glogovetan OE., Theoretical Aspects Regarding Artificial Inteligence Algotiyhms Used in Serial Production Programming, Review of Management and Economic Engineering Vol. 20, No. 4(82), December 2021, p. 252 - 258, ISSN (print): 1583-624X, ISSN (online): 2360-2155 Copyright © Association of Managers and Economic Engineers from Romania, Disponibil la: http://rmee.org/abstracturi/82/01_Articol_625_Lungu%20_ok.pdf

A3. LUCRĂRI INDEXATE ESCI:

Teodorescu M., Lungu F., Vlad R., “Improving Efficiency in Mass Production: Applying Genetic Algorithms in Conjunction with Objective Functions”, Acta Technica Napocensis, seria: Aplied Mathematics, Mechanics, and Engineering, vol67, Issue III ISSN 1221-5872L, Sept 2024 (în curs de indexare CLARIVATE).