**UNIVERSITATEA “ALEXANDRU IOAN CUZA” DIN IAȘI**

**FACULTATEA DE INFORMATICĂ**

****

LUCRARE DE LICENȚĂ

**Rețele de sortare de adâncime minimă**

**propusă de**

***Vlad Corjuc***

**Sesiunea: Iulie, 2021**

**Coordonator științific**

Lector Doctor Cristian Frăsinaru

**UNIVERSITATEA “ALEXANDRU IOAN CUZA” DIN IAȘI**

**FACULTATEA DE INFORMATICĂ**

Rețele de sortare de adâncime minimă

*Vlad Corjuc*

**Sesiunea:** *Iulie, 2021*

**Coordonator științific**

*Lector Doctor Cristian Frăsinaru*

Avizat,

Îndrumător Lucrare de Licență

Titlul, Numele și prenumele \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Data \_\_\_\_\_\_\_\_\_\_\_\_ Semnătura \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**DECLARAȚIE privind originalitatea conținutului lucrării de licență**

Subsemnatul(a) ………………………………………………………………………………………

domiciliul în …………………………………………………………………………………………………..

născut(ă) la data de ………………..…., identificat prin CNP ………….……………..………………..., absolvent(a) al(a) Universității „Alexandru Ioan Cuza” din Iași, Facultatea de ………………………. specializarea …………………………………………………………, promoția …………………………., declar pe propria răspundere, cunoscând consecințele falsului în declarații în sensul art. 326 din Noul Cod Penal și dispozițiile Legii Educației Naționale nr. 1/2011 art.143 al. 4 si 5 referitoare la plagiat, că lucrarea de licență cu titlul: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_elaborată sub îndrumarea dl. / d-na \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_, pe care urmează să o susțină în fața comisiei este originală, îmi aparține și îmi asum conținutul său în întregime.

De asemenea, declar că sunt de acord ca lucrarea mea de licență să fie verificată prin orice modalitate legală pentru confirmarea originalității, consimțind inclusiv la introducerea conținutului său într-o bază de date în acest scop.

Am luat la cunoștință despre faptul că este interzisă comercializarea de lucrări științifice in vederea facilitării falsificării de către cumpărător a calității de autor al unei lucrări de licență, de diploma sau de disertație și în acest sens, declar pe proprie răspundere că lucrarea de față nu a fost copiată ci reprezintă rodul cercetării pe care am întreprins-o.

Dată azi, ………………………… Semnătură student …………………………

DECLARAȚIE DE CONSIMȚĂMÂNT

Prin prezenta declar că sunt de acord ca Lucrarea de licență cu titlul „*Titlul complet al lucrării*”, codul sursă al programelor și celelalte conținuturi (grafice, multimedia, date de testetc.) care însoțesc această lucrare să fie utilizate în cadrul Facultății de Informatică.

De asemenea, sunt de acord ca Facultatea de Informatică de la Universitatea „Alexandru Ioan Cuza” din Iași, să utilizeze, modifice, reproducă și să distribuie în scopuri necomerciale programele-calculator, format executabil și sursă, realizate de mine în cadrul prezentei lucrări de licență.

Iași, *data*

Absolvent *Prenume Nume*

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(semnătura în original)

Cuprins

**Introducere**

Unul din cele mai studiate subiecte încă de la nașterea primelor calculatoare a fost *sortarea*, definită de Knuth D. E.ca o operație vitală nu doar în informatică, ci și în viața cotidiană [1, p. 1]. Sortarea unei grupări de elemente pe baza unui criteriu stabilit a avut o importanță deosebită datorită aplicabilității acesteia în majotitatea domeniilor.

În decursul anilor au fost prezentați numeroși algoritmi de sortare precum și structuri de date specifice pentru această operație. Astfel, una din primele structuri de date prezentate a fost *rețeaua de sortare*. Rețelele de sortare sunt structuri de date simple, bazate pe o serie de operații elementare (compararea și interschimbarea a două elemente), dar care au devenit centrul atenției multor cercetători interesați de studierea metodelor de sortare.

În această lucrare am abordat una din principalele probleme prezentată în studiul acestor structuri : *căutarea unei rețele de sortare de adâncime optimă*.

Alegerea temei de față a fost rezultatul curiozității mele cu privire la această metodă de sortare atât de simplă, dar cu un suport teoretic atât de vast, precum și rezultatul provocării pe care o lansează problema propusă, o dificultate care este încă nerezolvată.

În soluționarea acestei probleme am utilizat două metode distincte bazate pe o clasă de algoritmi de optimizare inspirați din procesul evolutiv. Prima metodă este formată dintr-un algoritm genetic hibrid care utilizează diverse operații de optimizare locală. Cea de a doua metodă este reprezentată de un algoritm genetic multi-obiectiv în care o importanță majoră este oferită factorului de noutate.

Prin această lucrare am dorit sa ofer o soluție atât teoretică cât și practică asupra problemei căutării unei rețele de sortare de adâncime optimă folosind algoritmi dintr-o clasă de algoritmi de optimizare inspirați din procesul evolutiv.

**Capitolul 1 – Rețele de sortare**

* 1. **Concepte de bază privind rețelele de sortare**

Rețelele de sortare fac parte dintr-o clasă specială a algoritmilor de sortare prin comparare, independent de tipul datelor, ce sortează un număr fix de elemente. Rețelele de sortare adaugă această restricție a numărului fix de elemente, pe când algoritmii de sortare prin comparare, de obicei, sortează orice număr de elemente [1, p. 219].

O rețea de sortare este formată din două componente : *comparatori* și *fire de date.*

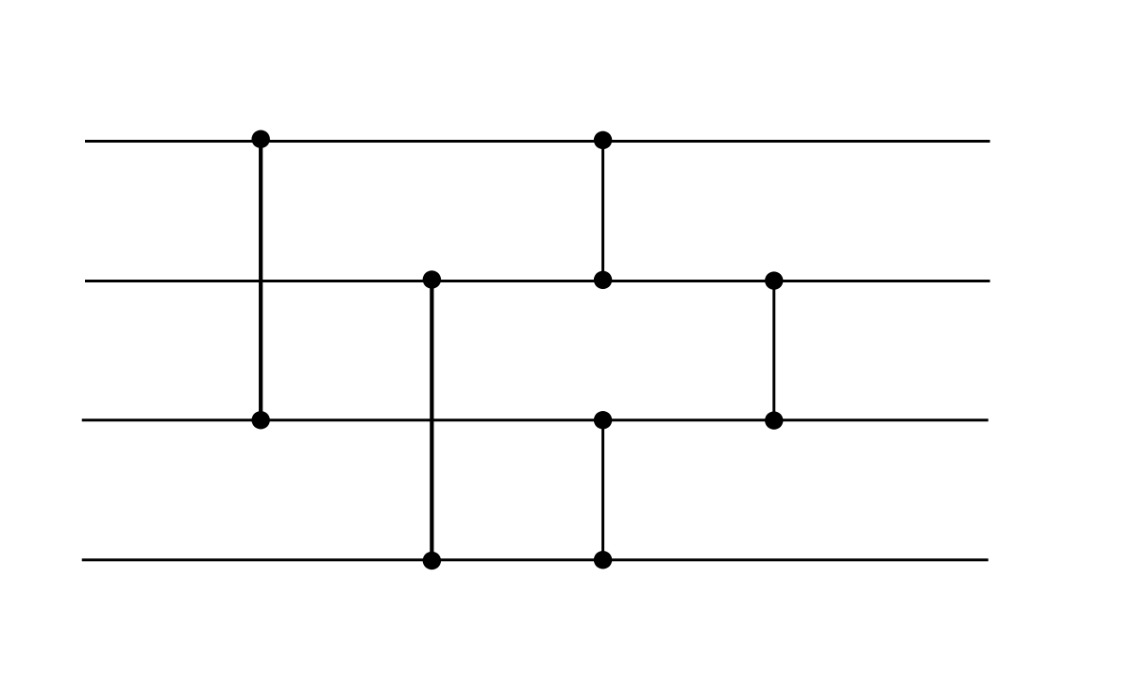
Numărul *firelor* esteegal cu cel al elementelor de sortat, ele conținând, la fiecare pas, câte un element din șirul inițial (de sortat). Pentru o înțelegere mai clară putem considera *firele* asemenea unor canale de date ce parcurg rețeaua de sortare de la stânga la dreapta, precum se poate observa în Figura 1.1.

Fiecare *comparator* va conecta două dintre *firele* rețelei. Aceștia vor compara și inversa elementele conținute de *firele* care le poartă astfel încât elementul cu valoarea minimă să fie pe *firul* superior iar cel cu valoarea maximă pe *firul* inferior*.*

În practică vom reprezenta *firele* utilizând linii orizontale iar comparatorii prin linii verticale între două fire. Pentru a evidenția conexiunea pe care o face un comparator între două *fire* vom utiliza mici cerculețe negre. Un exemplu de rețea de sortare este cel prezentat în Figura 1.1, pe când un exemplu de funcționare al rețelei se poate observa în Figura 1.2.

*fir*

*comparator*

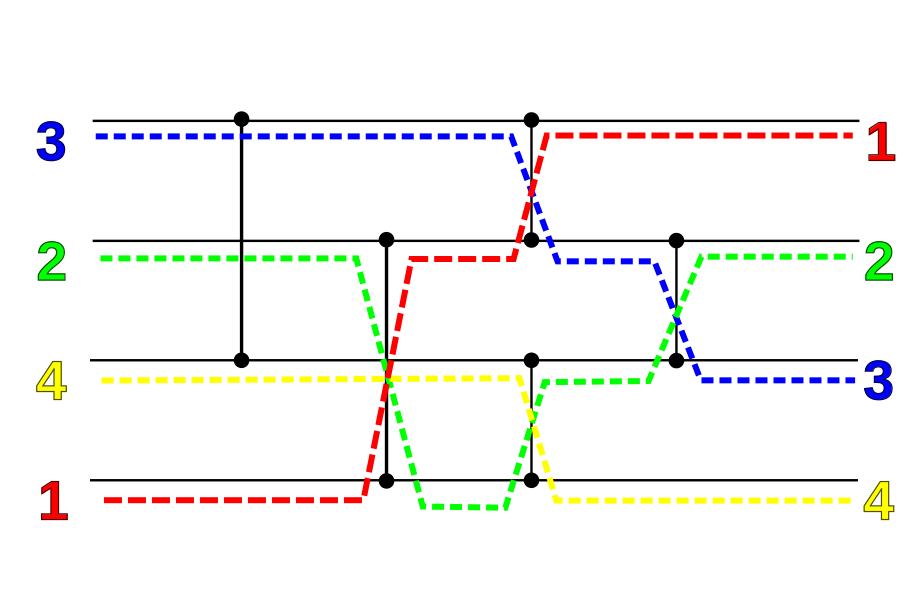


**Figura 1.1** : Rețea de sortare

**Figura 1.2** : Funționarea unei rețele de sortare

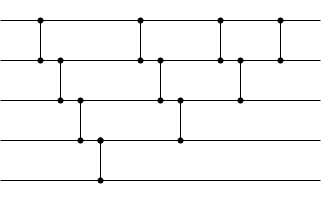
*comparator*

*fir*



Rețelele de sortare pot reprezenta orice algoritm de sortare bazat pe comparare și care este independent de tipul de date sortate.

În Figura 1.3 este prezentat algoritmului *Bubble Sort* (în cazul sortării unui șir de cinci elemente) sub forma unei rețele de sortare cu cinci fire și un număr de comparatori egal cu numărul comparărilor efectuate de algoritmul în cauză. O diferență conceptuală între cele două sortări se poate observa la fiecare iterație: algoritmul Bubble Sort va aduce elementul cel mai mare „la suprafață” pe când rețeaua de sortare prezentată va aduce elementul cel mai mare cât mai jos posibil, pe firul cel mai inferior.



**Figura 1.3**: Rețea de sortare reprezentând algoritmul Bubble Sort

* 1. **Aplicații ale rețelelor de sortare**

Rețelele de sortare pot fi implementate ușor fie în hardware, fie în software.

Rețelele de sortare sunt utilizate pentru a codifica constrângeri de tip *at-most-k* în SAT deoarece pot fi simulate în logica propozițională [2]. Astfel de constrângeri asigură faptul că cel mult k dintr-un set de variabile se află într-un set satisfăcător.

Batcher, în 1968, a sugerat utilizarea acestora pentru a construi rețele de comutare pentru hardware-ul computerului [3].

Din anii 2000, rețelele de sortare sunt utilizate de comunitatea GPGPU pentru construirea algoritmilor de sortare care să ruleze pe GPU.

Un alt exemplu al utilizării acestora este construirea unui filtru median pentru procesarea de imagini, utilizând câte o rețea de sortare pentru fiecare pixel care sortează pixeli vecini în funcție de intensitatea acestora [4, p. 8].

**Capitolul 2 – Rețele de sortare de adâncime minimă**

* 1. **Prezentarea problemei**

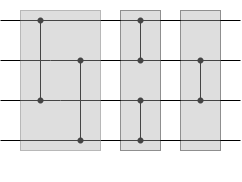
Încă de la introducerea rețelelor de sortare, cercetătorii au căutat rețele cu un număr minim de comparatoare astfel încât un prin criteriu pentru optimalitatea unei rețele a fost *dimensiunea* acestei: numărul de comparatori.

Ulterior s-a realizat posibilitatea aplicării în paralel a unui grup de comparatori (comparatori ce funcționează pe fire diferite). Un grup de comparatori consecutivi ce pot să fie aplicați în paralel se numește *strat*. Astfel a fost introdus un alt criteriu prin care putem analiza optimalitatea unei rețele de sortare, *adâncimea* (numărul de straturi).

În Figura 2.1 sunt exemplificate aceste concepte noi asupra rețelei de sortare cu patru fire prezentate anterior.

**Figura 2.1**: Rețea de sortare de dimensiune 5 și adâncime 3

*strat*



În acest cadru au apărut două probleme de optimizare considerând dimensiunea și adâncimea unei rețele de sortare:

1. Care este dimensiunea minimă a unei rețele de sortare cu *n* fire? Sau echivalent, care este numărul minim de comparări necesare pentru a sorta un șir cu *n* elemente?

2. Care este adâncimea minimă a unei rețele de sortare cu *n* fire? Sau echivalent, care este numărul minim de operații paralele necesare pentru a sorta un șir cu *n* elemente unde fiecare operație este formată dintr-un număr de comparări ?

Aceste probleme pot să fie reduse în timp polinomial (demonstrat ulterior) la două probleme de decizie:

1. Există o rețea de sortare cu *n* fire și de dimensiune *k?*
2. Există o rețea de sortare cu *n* fire și adâncime *d?*

Reducerea problemelor de optimizare la cele prezentate se poate realiza cu ușurință. Putem observa că odată cu creșterea numărului de fire, dimensiunea și adâncimea minimă nu pot să scadă : Având o rețea optimală (din punct de vedere al dimensiunii sau al adâncimii) cu *n+1* fire, putem construi o rețea de sortare cu *n* fire eliminând ultimul fir, se poate observa că aceasta va avea dimensiunea și adâncimea cel mult egale cu cele ale rețelei cu *n+1* fire.

Utilizând afirmația anterioară putem transforma problemele de optimizare în probleme de decizie utilizând un algoritm de căutare precum *căutarea binară.*

În cadrul acestei lucrări am analizat problema determinării unei rețele de sortare cu *n* fire și adâncime minimă.

**2.2 Complexitatea problemei**

Pentru a determina complexitatea problemei putem utiliza o metodă non-deterministă: găsim non-determinist o rețea cu n fire și de adâncime d și verificăm dacă aceasta sortează orice șir de *n* elemente. În urma considerației putem eticheta problema inclusă în NP. Însă această considerație a fost respinsă de rezultatele prezentate de Ian Parberry. Acesta a demonstrat că problema verificării faptului că o anumită rețea este o rețea de sortare este co-NP complete chiar și pentru rețelele de sortare cu adâncimea doar de mai mare decât adâncimea optimă [5].

Utilizând un oracol co-NP putem clasifica problema de optimizare a rețelelor de sortare din punct de vedere al adâncimii în clasa de complexitate . Deoarece putem transforma problema de optimizare în problema de existență prezentată anterior în timp polinomial, putem încadra problema existenței unei rețele de sortare cu n fire și de adâncime d în aceiași clasă de complexitate.

**2.3 Lucrări Conexe**

# **Bibliografie**

|  |  |
| --- | --- |
| [1] | D. E. Knuth, în *The Art of Computer Programming, Volume 3: (2nd Ed.) Sorting and Searching*, Redwood City, CA, USA, Addison Wesley Longman Publishing, 1998. |
| [2] | T. P. a. P. Steinke, „Pblib – a library for encoding pseudo-boolean constraints into cnf,” în *Theory and Applications of Satisfiability Testing – SAT 2015, ser. Lecture Notes in Computer Science, M. Heule and S. Weave*, vol. 9340, Eds. Springer International Publishing, 2015, pp. 9-16. |
| [3] | K. E. Batcher, „Sorting networks and their applications,” în *AFIPS Spring Joint Computer Conference*, 1968. |
| [4] | T. Haslop, „Minimal Depth Sorting Networks,” March 2020. [Interactiv]. Available: https://www.szi.uni-bremen.de/wp-content/uploads/2020/03/thesis\_compressed.pdf. |
| [5] | I. Parberry, „On the Computational Complexity of Optimal Sorting Network,” 1991. [Interactiv]. Available: http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.52.9568. |
| [6] | M. Budiu, „Maşini alternante, ierarhia polinomială,” 16 noiembrie 1999. [Interactiv]. Available: https://www.cs.cmu.edu/~mihaib/articole/complex/complex-html.html#SECTION03070000000000000000. |