Temă Seminar 3

Adrian-Octavian Pătrașcu

Noiembrie 2022

1 Exercițiul 6

Graful minim conex pentru care $T_{DFS} = T_{BFS}$ este un arbore. Se observă ușor că indiferent ce rădăcină alegem, parcurgerile vor genera același arbore.
Astfel, rămâne de investigat cazul în care avem cel puțin un ciclu în graf în care putem întâlni un ciclu (v_1, v_2, ..., v_k, v_1) care in T_{DFS} ar fi un lanț de la v_1 la v_k, iar în T_{BFS} v_k ar fi pe același nivel cu v_2, garantând $T_{DFS} \neq T_{BFS}$.

2 Exercițiul 7

V = multimea intervalelor $[a_i, b_i]$, i = $\overline{1, n}$
E = multimea muchiilor (u,v,c) în care u și v nu se intersectează, iar c este suma lungimilor segmentelor u si v.
Parcurgem cu Dijkstra din primele **depth** noduri, unde **depth** este adancimea intervalelor. Timp: O(depth * $O(mlog(n))$)
Din cauza lipsei de timp și a întinderii soluției, las o resursă către o soluție în timp liniar a problemei la [3].

3 Exercițiul 8

Metoda 1
Este cunoscut faptul că, dându-se N noduri, avem $2^{\binom{N}{2}}$ grafuri partiale neorientate. Dintre acestea, avem nevoie sa verificam toate muchiile posibile în graf pentru a vedea în ce configurație ne aflăm, lucru care necesită $\binom{N}{2}$ query-uri. Prin configurație se înțelege G = (V,E) din care putem afla ușor numarul de componente conexe.
Metoda 2
După orice query, numărul nostru de componente conexe, notat cu CoCo, o să fie mai

mic sau egal cu cel dinaintea query-ului respectiv. Astfel, să presupunem că facem $\binom{N}{2}$ - 1 query-uri din care rezultă că avem N Coco. Există o șansă ca al $\binom{N}{2}$-lea query să găsească o muchie în graf și astfel să scadă numarul de CoCo.

4 Exercițiul 9

Problema în discuție este o formulare a Travelling Salesman (TSP). Formal, fie G = (V,E) $\in K_n$. Cautam o permutare ciclica a nodurilor lui G astfel încât suma distanțelor dintre noduri să fie minimă. Notăm permutarea ciclică soluție cu C_{P_n}
Țin să menționez trivialitatea insuficienței unei simple parcurgeri a grafului, deoarece până și o transpoziție în ciclul găsit poate schimba considerabil suma finală într-una mai mică (explicația este lăsată spre deducere cititorului).

Soluția 1
O primă abordare cuprinde parcurgerea tuturor ciclurilor și determinarea celui de sumă minimă. Complexitatea acesteia este $O((n-1)!)$, deoarece întotdeauna știm nodul de început.

Soluția 2
O euristică categoric favorabilă în fața soluției brute de mai sus este algoritmul Nearest Neighbor care adaugă la nodurile parcurse până la pasul k cel mai apropiat nod de nodul k, repetând procesul până la completarea ciclului. Complexitatea NN-ului este de $O(N^2)$.

Soluția 3
O altă euristică constă în unirea celor mai apropiate doua noduri curente fară a închide ciclu prematur. Complexitatea soluției este $O(n^2log(n))$ datorita sortarii muchiilor.

Observație pentru soluțiile 2 și 3
În practică, cele două euristici singure sau împreună pot produce rezultate dezamăgitor de îndepărtate de limita inferioară teoretică și ar trebui interpretate în continuare numai drept euristici.

Soluția 3
În mod convenabil, APM-ul grafului G seamană destul de mult cu ciclul soluției, lucru de care am putea să ne folosim. Similaritatea grafurilor se trage din faptul că subgraful format din nodurile de grad par o să facă parte din C_{P_n}.
Ne dorim însă să aducem toate nodurile la grad par. Pentru aceasta o să luam G‘=(V,E,F), unde F = (u,v,c), u, v $\in$ G, iar c = cost(u,v) este minim. Pentru a

genera multimea F de muchii, o să luăm iterăm printre toate perechile de noduri de grad impar și vom selecta configurația de cost total minim.
Pe G' facem un tur eulerian din orice nod și reținem nodurile parcurse. În final vom avea ciclul soluție. Complexitate finală: $O(n^4)$ din cauza căutarii partiției optime pentru F.
Se garantează că soluția 4 poate fi de 1.5 ori mai lungă decât optimul.[7]

References

[1] Dieter Jungnickel, *"Graphs, Networks and Algorithms Second Edition."*

[2] Wikipedia - Interval Graph.

[3] Shuchi Chawla, CS787: Advanced Algorithms, Lecture 3: Dynamic Programming, (https://pages.cs.wisc.edu/shuchi/courses/787-F09/scribe-notes/lec3.pdf), University Of Wisconsin.

[4] Atallah, Mikhail J.; Chen, Danny Z.; and Lee, D. T., "An Optimal Algorithm for Shortest Paths on Weighted Interval and Circular-Arc Graphs with Applications" (1993). Department of Computer Science Technical Reports. Paper 1024.

[5] Gregory Gutin, Anders Yeo, Alexey Zverovich, Traveling salesman should not be greedy: domination analysis of greedy-type heuristics for the TSP, Discrete Applied MathematicsVolume 117, Issues 1–3, 2002, Pages 81-86.

[6] Jørgen Bang-Jensen, Gregory Gutin, Anders Yeo, When the greedy algorithm fails, Discrete Optimization, Volume 1, Issue 2, 2004, Pages 121-127.

[7] David S. Johnson, Lyle A. McGeoch, The Traveling Salesman Problem: A Case Study in Local Optimization