UNIUNEA EUROPEANĂ

GUVERNUL ROMÂNIEI

Instrumente Structurale
2007-2013

Platformă de e-learning și curriculă e-content pentru învățământul superior tehnic

Proiectarea Algoritmilor

20. Arbori minimi de acoperire

Bibliografie

[1] http://monalisa.cacr.caltech.edu/monalisa__Service_Applications__Monitoring_VRVS.html

[2] http://www.cobblestoneconcepts.com/ucgis2summer2002/guo/guo.html

[3] Giumale – Introducere in Analiza Algoritmilor cap. 5.5

[4] R. Sedgewick, K Wayne – curs de algoritmi Princeton 2007 www.cs.princeton.edu/~rs/AlgsDS07/ 01UnionFind si 14MST

[5] http://www.pui.ch/phred/automated_tag_clustering/

[6] Cormen – Introducere în Algoritmi cap. 24

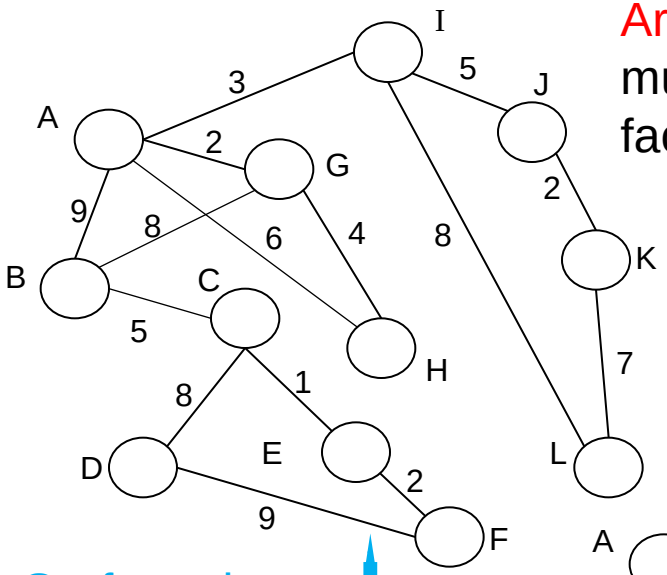
Planul cursului

- Arbori minimi de acoperire:
  - Definiție;
  - Utilizare;
  - Algoritmi.
- Operații cu mulțimi disjuncte:
  - Structuri de date pentru reprezentarea mulțimilor disjuncte;
  - Algoritmi pentru reuniune si căutare;
  - Calcul de complexitate.

Arbori minimi de acoperire – Definiții

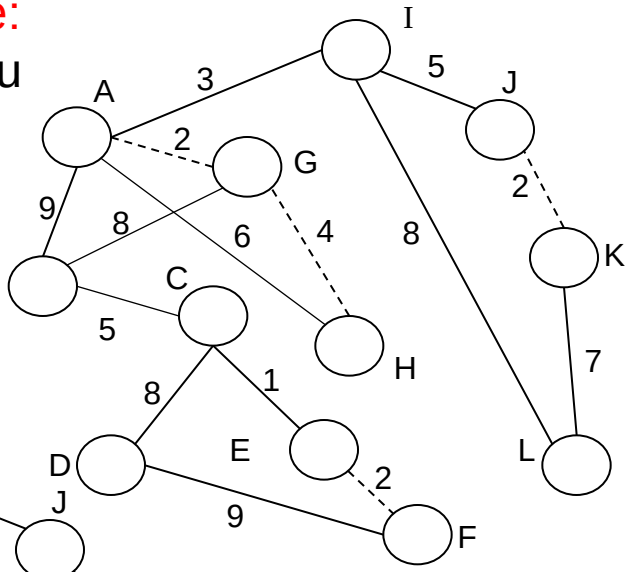
- Fie G = (V,E) graf neorientat si conex, iar w: E → $\mathfrak{R}$ o funcție de cost (w(u,v) = costul muchiei (u,v)).
- Definiție: Un arbore liber al lui G este un graf neorientat conex si aciclic Arb = (V',E'); V' $\subseteq$ V, E' $\subseteq$ E. Costul arborelui este: C(Arb) = Σ w(e), e $\in$ E'.
- Definiție: Un arbore liber se numește arbore de acoperire dacă V' = V.
- Definiție: Un arbore de acoperire (Arb) se numește arbore minim de acoperire (notăm AMA) dacă Arb $\in$ ARB(G) a.i. C(Arb) = min{C(Arb') | Arb' $\in$ ARB(G)}.

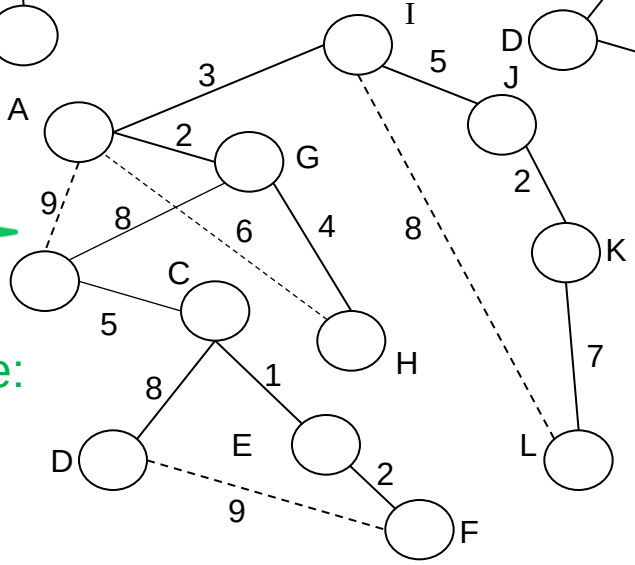
Exemple



Graf neorientat conex

Arbore de acoperire:
muchiile punctate nu fac parte din arbore



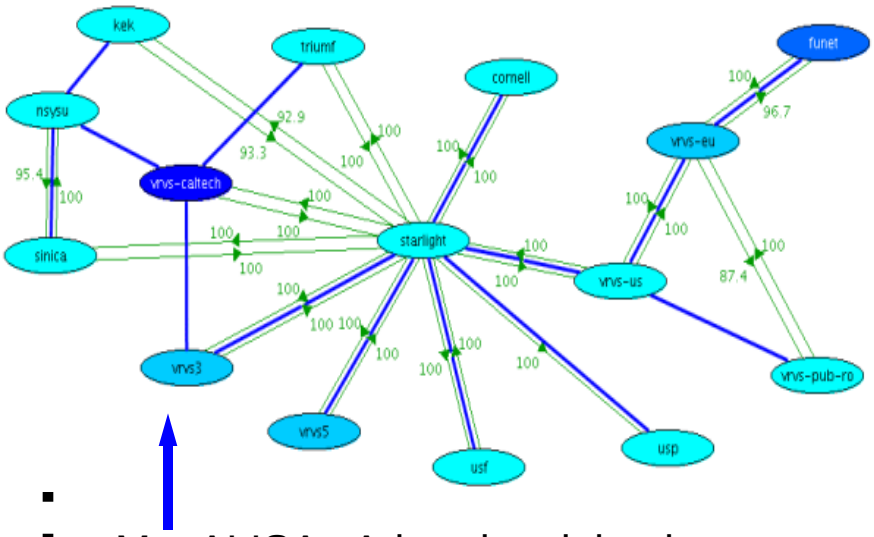


Arbore minim de acoperire:
muchiile punctate au fost eliminate din graf

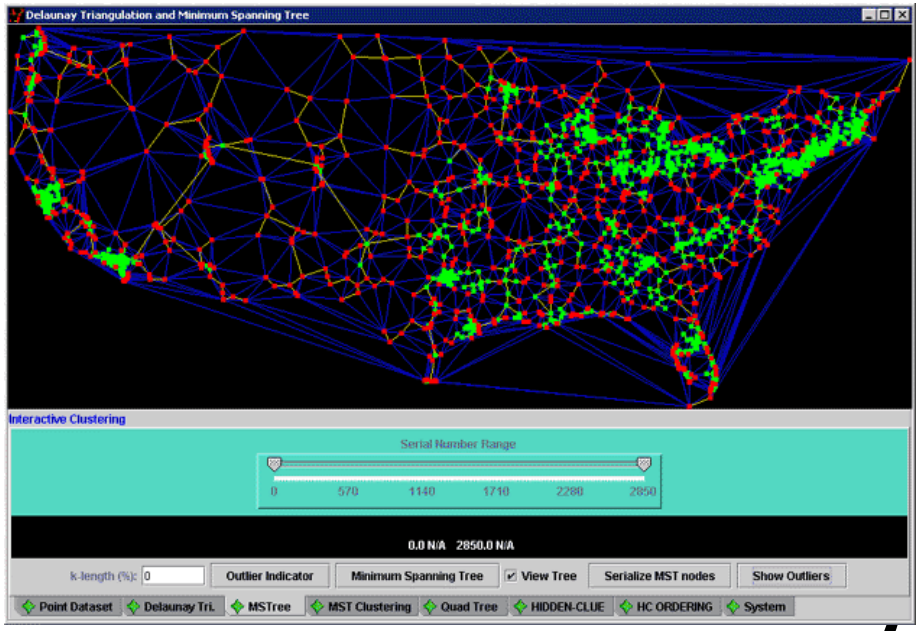
Utilizări

- Proiectarea rețelelor:
  - Electrice, calculatoare, drumuri.
- Clustering.
- Algoritmi de aproximare pentru probleme NP-complete.

Exemple de utilizare



MonALISA - Arborele minim de acoperire al conexiunilor si calitatea conexiunilor peer-to-peer pentru un set de relee VRVS (caltech) [1]



Arbore minim de acoperire pentru cca 2850 de orase din USA [2]

AMA – Definiții (II)

- Definiție: Fie $A \subseteq E$ o mulțime de muchii ale unui graf $G = (V,E)$ și $(S,V-S)$ o partiționare a lui V. Partiționarea respectă mulțimea A dacă $\nexists e \in A$ care taie frontiera dintre S și $V-S$ ($\forall (u,v) \in A \rightarrow u,v \in S$ sau $u,v \in V-S$).

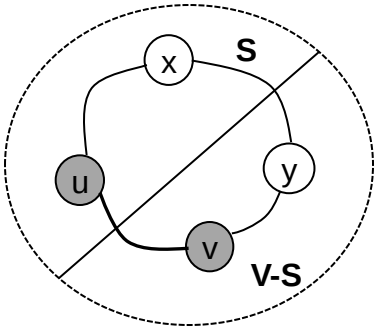
- Definiție: Fie $A \subseteq E'$ o mulțime de muchii ale unui AMA $Arb = (V,E')$ al grafului $G = (V,E)$, iar $e \in E$ o muchie oarecare din G. Muchia e este sigură in raport cu A dacă mulțimea $A \cup \{e\}$ face parte dintr-un AMA al lui G.

AMA – Teorema

- Teorema 5.23: Fie A o mulțime de muchii ale unui AMA al grafului G = (V,E). Fie (S,V-S) o partiționare care respectă A, iar (u,v) $\in$ E o muchie care taie frontiera dintre S si V-S a.î.

w(u,v) = min{w(x,y) | (x,y) $\in$ E & (x $\in$ S, y $\in$ V-S) or (x $\in$ V-S, y $\in$ S)}.

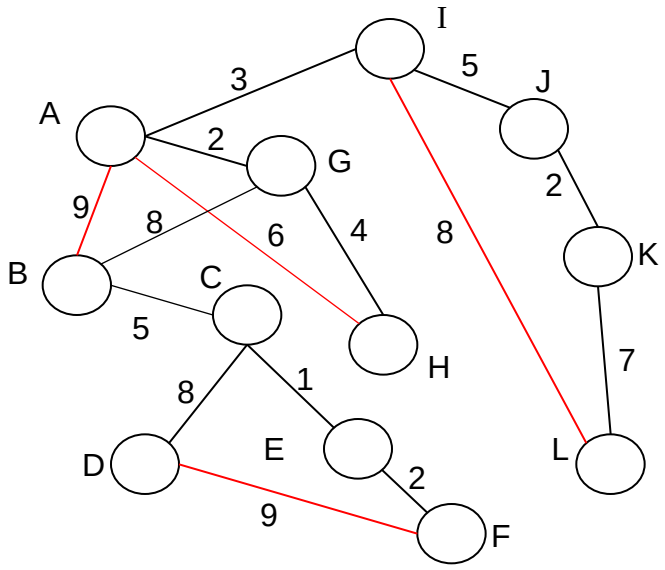
Muchia (u,v) este sigură in raport cu A.



- Dem (Reducere la absurd):
  - pp (u,v) nu e muchie sigură.
  - (I) → ∃ AMA Arb' = (V,E'), a.i. A ⊆ E'. Pp (u,v) ∉ Arb'
  - In Arb' ∃ cale u..v → ∃ (x,y) ∈ u..v care taie partiționarea si (x,y) ∈ Arb'
  - (x,y) ∉A, (u,v) ∉ A pt. ca partiționarea respecta A, iar w(u,v) ≤ w(x,y) (I)
  - Dacă in Arb' eliminăm (x,y) si adăugăm (u,v) → Arb" = (V,E"), E" = E' – {(x,y)} + {(u,v)}
  - C(Arb") ≤ C(Arb'), Arb' – AMA → C(Arb') = C(Arb") → Arb" – AMA → (u,v) – muchie sigura.

Proprietăți (I)

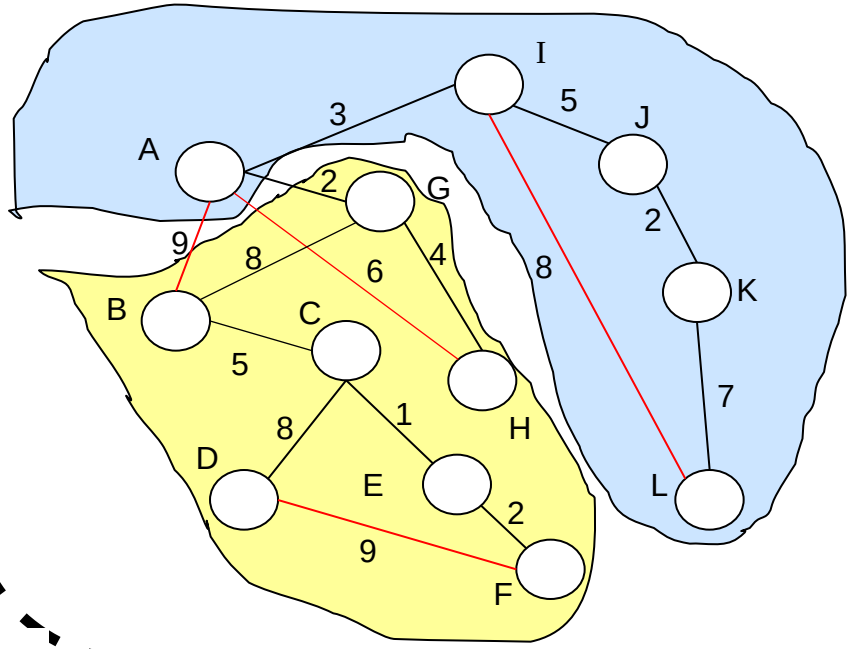
G = (V,E), C = (V',E') – ciclu in G; e ∈ E' a.î. w(e) = max {w(e') | e' ∈ E'} => e ∉ Arb(G) unde Arb(G) = AMA in G.



- Dem (Reducere la absurd): Pp e ∈ Arb(G).
- Eliminând e din Arb(G) → 2 mulțimi de muchii: S_1, S_2.
- e ∈ E' (ciclu) → ∃ e' ∈ E', w(e) > w(e') a.î. un capăt din e' este in S_1 și celalalt in S_2.
- Arb(G) – e + e' = arbore de acoperire.
- Cost(Arb(G) - w(e) + w(e') < Cost(Arb(G)) => Arb(G) nu este arbore minim.

Proprietăți (II)

G = (V,E), S = (V',E'), V' $\subset$ V; e = (u,v) a.î. e $\notin$ E' și (u $\in$ V' si v $\notin$ V') sau (u $\notin$ V' si v $\in$ V') cu proprietatea că:
w(u,v) = min{w(u',v')| (u'$\in$V' si v'$\notin$V') sau (u'$\notin$V' si v'$\in$V')} => (u,v) $\in$ AMA.



•Dem (Reducere la absurd): Pp e $\notin$ Arb(G).

•Arb' = Arb(G) – e' + e (unde e' o muchie similară cu e).

•Arb'= arbore de acoperire.

•Cost(Arb') < Cost(Arb) → Arb nu este arbore minim.

AMA

- Bazați pe ideea de muchie sigură – se identifică o muchie sigură si se adaugă in AMA.

- 2 algoritmi de tip greedy:
  - Prim: se pornește cu un nod si se extinde pe rând cu muchiile cele mai ieftine care au un singur capăt in mulțimea de muchii deja formată (Proprietatea 2). Algoritmul este asemănător algoritmului Dijkstra.

  - Kruskal: inițial toate nodurile formează cate o mulțime si la fiecare pas se reunesc 2 mulțimi printr-o muchie. Muchiile sunt considerate in ordinea costurilor si sunt adăugate in arbore dacă nu creează ciclu (Proprietatea 1).