Q0: De ce folosim algoritmi aproximativi?

* Problemele din NP-hard (cel putin la fel de dificile ca cele din NP-C) nu au algoritmi fezabili (in timp polinomial) pt determinarea optimului. Asa ca una dintre solutiile de compromis ar fi sa gasim o solutie “aproape” optima.

Q1: ce este factorul de aproximare pentru un algoritm?

Fie ALG - solutia noastra, OPT - solutia optima

In cazul unei probleme de minim, o constanta *c* se numeste factor de aproximare daca:

OPT≤ALG≤c\*OPT.

in cazul unei probleme de maxim

OPT≥ALG≥c\*OPT.

Q1.1 In cazul unei probleme de minim: Un algoritm 2-aproximativ poate fi numit 3-aproximativ?

Consecinta: pt un ALG sa gasim *c* - cat mai aproape de 1 (tight bound)

Q1.2 Cum putem sa justidicam ca un *c* dat este tight bound?

In cazul problemelor de minim, ar tb sa gasim o intrare *I* astfel incat:

ALG(I) **=** c\*OPT(I)

Probleme:  
1. Avem următorul scenariu: Avem *n* colete de transportat, fiecare avand greutatea de *w1, w2,...,wn.* Pentru a le transporta, putem folosi un număr de camioane, fiecare avand capacitatea de transport *G*. Presupunem că *wi≤G*, pentru orice *i*. Ne dorim sa minimizăm numărul de camioane folosite. Considerăm următorul plan de încărcare a camioanelor:

Odată ce avem la dispoziție un camion pt a fi încărcat, iterăm prin mulțimea coletelor, incărcându-le in camion, până când dăm peste primul colet ce nu mai incape. În acel moment considerat că am terminat de încărcat camionul curent și trecem la următorul camion, prima dată încărcând coletul care nu a mai încăput în cel precedent.

1. Arătați, printr-un exemplu simplu, că metoda de mai sus nu furnizează soluția optimă.
2. Arătați totuși că soluția de mai sus este un algoritm 2-aproximativ pentru problema noastră.

Rezolvare:

1. G=50; w={10,40,20,35,30,15}

OPT=(10,40);(20,30);(35,15)

ALG=(10,40); (20); (35);(30,15)

1. fie *i* primul obiect care nu incape in camionul care contine deja incarcatura Tj.

(Care ar fi un lower bound - un numar absolut minim de camioane - necesar pentru transport? 1/G \* ∑(wi) - fiecare camion sa fie incarcat la limita maxima; deci OPT>= 1/G \* ∑(wi))

Pentru a transporta Tj si obiectul i folosim 2 camioane;

dar cat e w(Tj)+w(i)>G

Numarul de camioane folosit: ALG<=2\*1/G \* ∑(wi)<=2\*OPT (la doua caminoane folosite, cu siguranta transport o greutate >G)

2) Dat fiind algoritmul Load-Balance (Cursul 2, slide 19) să se stabilească dacă următoarea afirmație este adevărată sau falsă.

”Pentru orice instanța a problemei de Load-Balace, exista o anumită ordine a procesării activităților astfel încât algoritmul de tip greedy să dea o soluție optimă”

Dacă afirmația este adevărată, oferiți o demonstrație, altfel, găsiți un contraexemplu.

Raspuns: Da! Daca am avea acces la solutia optima, am sti ce activitate este pe fiecare procesor. Atunci, la un anumit pas intermediar am sti ce procesor j este cu incarcatura cea mai mica, inductiv ne-am asigura ca acel procesor va primi in momentul respectiv fix activitatea optima pt el.

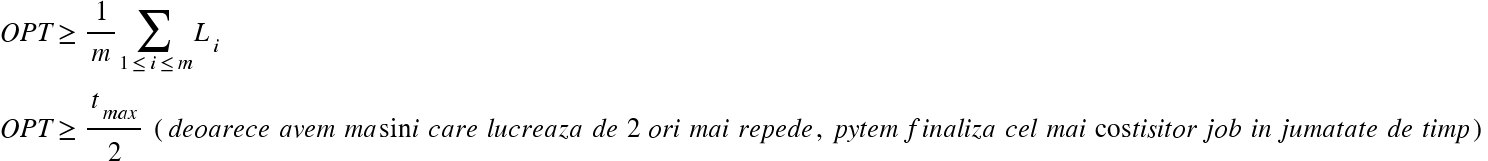
Observatie: Problema incarcarii optime (load balance) este de fapt o problema de permutare!

3) Fie Problema Load Balance, dar cu următoarea modificare: Avem *n* joburi si *m* mașini, doar că pentru primele *k* mașini timpul de lucru al unei activități este înjumătățit. Să se găsească un algoritm bazat pe tehnica greedy care furnizeaza o soluție de cel mult 3xOPT.

Raspuns:

Algoritmul problemei noastre este exact ca cel din curs (slide 19) cu observatia ca la linia 5 avem i<=k (msina selectat se aflta printre cele care lucreaza de doua ori mai repede) la linia 7 vom avea Li+=tj/2

Trebuie sa aratam ca acest algoritm este 3 aprox



<math xmlns="http://www.w3.org/1998/Math/MathML"><mi>f</mi><mi>i</mi><mi>e</mi><mo>&#xA0;</mo><mi>q</mi><mo>&#xA0;</mo><mo>-</mo><mo>&#xA0;</mo><mi>m</mi><mi>a</mi><mi>sin</mi><mi>a</mi><mo>&#xA0;</mo><mi>c</mi><mi>u</mi><mo>&#xA0;</mo><mi>i</mi><mi>n</mi><mi>c</mi><mi>a</mi><mi>r</mi><mi>c</mi><mi>a</mi><mi>t</mi><mi>u</mi><mi>r</mi><mi>a</mi><mo>&#xA0;</mo><mi>m</mi><mi>a</mi><mi>x</mi><mi>i</mi><mi>m</mi><mi>a</mi><mo>&#xA0;</mo><mi>l</mi><mi>a</mi><mo>&#xA0;</mo><mi>s</mi><mi>f</mi><mi>a</mi><mi>r</mi><mi>s</mi><mi>i</mi><mi>t</mi><mi>u</mi><mi>l</mi><mo>&#xA0;</mo><mi>a</mi><mi>l</mi><mi>g</mi><mi>o</mi><mi>r</mi><mi>i</mi><mi>t</mi><mi>m</mi><mi>u</mi><mi>l</mi><mi>u</mi><mi>i</mi><mo>&#xA0;</mo><mi>n</mi><mi>o</mi><mi>s</mi><mi>t</mi><mi>u</mi><mspace linebreak="newline"/><mi>d</mi><mi>e</mi><mi>c</mi><mi>i</mi><mo>&#xA0;</mo><mi>A</mi><mi>L</mi><mi>G</mi><mo>&#xA0;</mo><mo>=</mo><mo>&#xA0;</mo><mi>L</mi><mi>q</mi><mspace linebreak="newline"/><mi>F</mi><mi>i</mi><mi>e</mi><mo>&#xA0;</mo><mi>p</mi><mo>&#xA0;</mo><mo>-</mo><mo>&#xA0;</mo><mi>u</mi><mi>l</mi><mi>t</mi><mi>i</mi><mi>m</mi><mi>u</mi><mi>l</mi><mo>&#xA0;</mo><mi>j</mi><mi>o</mi><mi>b</mi><mo>&#xA0;</mo><mi>a</mi><mi>d</mi><mi>a</mi><mi>a</mi><mi>u</mi><mi>g</mi><mi>a</mi><mi>t</mi><mo>&#xA0;</mo><mi>l</mi><mi>a</mi><mo>&#xA0;</mo><mi>L</mi><mi>q</mi><mspace linebreak="newline"/><mi>f</mi><mi>i</mi><mi>e</mi><mo>&#xA0;</mo><mi>L</mi><mo>'</mo><mi>i</mi><mo>&#xA0;</mo><mo>-</mo><mo>&#xA0;</mo><mi>l</mi><mi>o</mi><mi>a</mi><mi>d</mi><mi>u</mi><mi>l</mi><mo>&#xA0;</mo><mi>m</mi><mi>a</mi><mi>sin</mi><mi>i</mi><mi>i</mi><mo>&#xA0;</mo><mo>'</mo><mi>i</mi><mo>'</mo><mo>&#xA0;</mo><mi>f</mi><mi>i</mi><mi>x</mi><mo>&#xA0;</mo><mi>i</mi><mi>n</mi><mi>a</mi><mi>i</mi><mi>n</mi><mi>t</mi><mi>e</mi><mo>&#xA0;</mo><mi>d</mi><mi>e</mi><mo>&#xA0;</mo><mi>a</mi><mo>&#xA0;</mo><mi>f</mi><mi>i</mi><mo>&#xA0;</mo><mi>a</mi><mi>d</mi><mi>a</mi><mi>u</mi><mi>g</mi><mi>a</mi><mi>t</mi><mi>a</mi><mo>&#xA0;</mo><mi>a</mi><mi>c</mi><mi>t</mi><mi>i</mi><mi>v</mi><mi>i</mi><mi>t</mi><mi>a</mi><mi>t</mi><mi>e</mi><mo>&#xA0;</mo><mi>p</mi><mo>&#xA0;</mo><mi>l</mi><mi>a</mi><mo>&#xA0;</mo><mi>m</mi><mi>a</mi><mi>sin</mi><mi>a</mi><mo>&#xA0;</mo><mi>q</mi><mo>.</mo><mspace linebreak="newline"/><mi>A</mi><mi>L</mi><mi>G</mi><mo>=</mo><mi>L</mi><mi>q</mi><mo>&#x2264;</mo><mi>L</mi><mo>'</mo><mi>q</mi><mo>+</mo><msub><mi>t</mi><mi>p</mi></msub><mo>&#xA0;</mo><mfenced><mrow><mi>d</mi><mi>e</mi><mi>o</mi><mi>a</mi><mi>r</mi><mi>e</mi><mi>c</mi><mi>e</mi><mo>&#xA0;</mo><mi>u</mi><mi>l</mi><mi>t</mi><mi>i</mi><mi>m</mi><mi>u</mi><mi>l</mi><mo>&#xA0;</mo><mi>t</mi><mi>e</mi><mi>r</mi><mi>m</mi><mi>e</mi><mi>n</mi><mo>&#xA0;</mo><mi>p</mi><mi>o</mi><mi>a</mi><mi>t</mi><mi>e</mi><mo>&#xA0;</mo><mi>f</mi><mi>i</mi><mo>&#xA0;</mo><mo>&#xA0;</mo><msub><mi>t</mi><mi>p</mi></msub><mo>&#xA0;</mo><mi>d</mi><mi>a</mi><mi>c</mi><mi>a</mi><mo>&#xA0;</mo><mi>q</mi><mo>&gt;</mo><mi>k</mi><mo>&#xA0;</mo><mi>s</mi><mi>a</mi><mi>u</mi><mo>&#xA0;</mo><msub><mi>t</mi><mi>p</mi></msub><mo>/</mo><mn>2</mn><mo>,</mo><mo>&#xA0;</mo><mi>a</mi><mi>l</mi><mi>t</mi><mi>f</mi><mi>e</mi><mi>l</mi></mrow></mfenced><mspace linebreak="newline"/><mo>&#xA0;</mo><mo>&#xA0;</mo><mo>&#xA0;</mo><mo>&#xA0;</mo><mo>&#xA0;</mo><mo>&#xA0;</mo><mo>&#xA0;</mo><mo>&#xA0;</mo><mo>&#xA0;</mo><mo>&#xA0;</mo><mo>&#xA0;</mo><mo>&#xA0;</mo><mo>&#xA0;</mo><mo>&#xA0;</mo><mo>&#x2264;</mo><mfrac><mn>1</mn><mi>m</mi></mfrac><munder><mo>&#x2211;</mo><mrow><mn>1</mn><mo>&#x2264;</mo><mi>i</mi><mo>&#x2264;</mo><mi>m</mi></mrow></munder><mi>L</mi><mo>'</mo><mi>i</mi><mo>&#xA0;</mo><mo>+</mo><msub><mi>t</mi><mrow><mi>m</mi><mi>a</mi><mi>x</mi></mrow></msub><mo>&#xA0;</mo><mo>&#x2264;</mo><mfrac><mn>1</mn><mi>m</mi></mfrac><munder><mo>&#x2211;</mo><mrow><mn>1</mn><mo>&#x2264;</mo><mi>i</mi><mo>&#x2264;</mo><mi>m</mi></mrow></munder><mi>L</mi><mi>i</mi><mo>&#xA0;</mo><mo>+</mo><msub><mi>t</mi><mrow><mi>m</mi><mi>a</mi><mi>x</mi></mrow></msub><mo>&#x2264;</mo><mi>O</mi><mi>P</mi><mi>T</mi><mo>+</mo><mn>2</mn><mo>&#xB7;</mo><mi>O</mi><mi>P</mi><mi>T</mi><mo>=</mo><mn>3</mn><mo>&#xB7;</mo><mi>O</mi><mi>P</mi><mi>T</mi></math>