Soluţie "Load Balancing with Restrictions"

Ştefan Popescu

Problemă

(15p) Se consideră un proiect format din n task-uri ce trebuiesc efectuate de către m mașini de calcul. Fiecare task poate fi procesat doar de către una din două mașini dintre cele m. Task-urile sunt caracterizate ca fiind un triplete de forma (T_i, x_i, y_i), unde T_i este timpul necesar pentru a procesa task-ul i, iar x_i și y_i sunt indicele celor două mașini ce pot efectua task-ul i (celelalte $m-2$ mașini sunt incompatibile cu efectuarea taskului i). Se dorește planificarea fiecărui task pe câte o mașină compatibilă cu task-ul astfel încât întregul proiect să se termine cât mai repede. (Altfel spus, se dorește minimizarea timpului de lucru a mașinii celei mai solicitate.)

Cerințe

a) Să se scrie problema anterioară sub forma unei ***Probleme de Programare Liniară cu Numere Întregi*** (en. *Integer Linear Programming Problem*). Apoi această problemă să fie *relaxată* și adusă sub forma unei ***Probleme de Programare Liniară***. **(10p)**

b) Folosindu-vă de Problemele de Programare Liniară descrise la punctul a), propuneți un algoritm 2-aproximativ pentru problema inițială. Justificați de ce algoritmul propus are factorul de aproximare 2. **(5p)**

Notații și indicații:

OPT - încărcătura mașinii celei mai solicitate în configurația optimă.

ALG - încărcătura mașinii celei mai solicitate în urma algoritmului propus de voi.

LP, respectiv ILP - expresiile ce trebuiesc maximizate sau minimizte pentru problemele voastre de programare liniară, respectiv programare liniară cu numere îtregi.

Task-urile vor fi indexate cu variabile de forma "i, j, k". Mașinile vor fi indexate cu variabile de forma "q, p, r".

$Comp(q)$ - lista task-urilor compatibile cu mașina q

Variabilele de tipul A_q^i vor indica dacă task-ul i este alocat mașinii q sau nu.

Soluţie

a) minimizați ILP cu proprietatea că:

- pentru fiecare mașină q avem: $\sum_{i \in Comp(q)} A_q^i * T_i \leq ILP$ (suma timpilor task-urilor alocate pe fiecare mașină nu poate depăși ILP)
- $A_{x_i}^i + A_{y_i}^i = 1$ pentru orice $i \in \{1, ..., n\}$ (Un task trebuie programat în totalitate folosindu-ne de cele două maşini compatibile cu el)[1]

[1]Mulți ar tinde să scrie $A_{x_i}^i + A_{y_i}^i \geq 1$. Deși această restricție va furniza o soluție cu load optim, este posibil ca o activitate să fie programată de două ori, deorece nu există o restricție.

* pentru fiecare $(i,q) \in \{1,...,n\} \times \{1,...,m\}$ $A_q^i = 0$ dacă $i \notin Comp(q)$ şi $A_q^i \in \{0,1\}$ altfel.[2] (taskurile pot fi programate, adică A_q^i poate fi nenul, doar pe maşinile compatibile)

minimizați LP cu proprietatea că:

* pentru fiecare mașină q avem: $\sum_{i \in Comp(q)} A_q^i * T_i \leq ILP$ (suma timpilor task-urilor alocate pe fiecare mașină nu poate depăși ILP)
* pentru fiecare $(i,q) \in \{1,...,n\} \times \{1,...,m\}$ $A_q^i = 0$ dacă $i \notin Comp(q)$ şi $0 \leq A_q^i \leq 1$ altfel. (taskurile pot fi programate, adică A_q^i poate fi nenul, doar pe maşinile compatibile)
* $A_{x_i}^i + A_{y_i}^i = 1$ pentru orice $i \in \{1,...,n\}$ (Un task trebuie programat în totalitate folosindu-ne de cele două maşini compatibile cu el)

b) Ştim că $LP \leq ILP = OPT$. Odată obținut LP putem face următoarea construcție: Pentru fiecare task i, alegem maximul dintre $A_{x_i}^i$ și $A_{y_i}^i$ și îl facem egal cu 1, respectiv pe celălalt îl facem egal cu 0. Valoarea maximă dintre cele două ($A_{x_i}^i$ și $A_{y_i}^i$) va fi măcar 1/2 deoarece suma lor este 1, asta conduce la faptul că încărcătura unei mașini în urma acestui artificiu va fi de cel mult de 2 ori mai mare decât iniţial. Adică avem $ALG \leq 2 * LP \leq 2 * ILP = 2 * OPT$.

[2] La o primă vedere această constrângere ar fi acoperită de prima, anume că o activitate trebuie să fie realizată dintre una din cele două mașini. Dar, teoretic, fără acestă restricție pot fi planificate și duplicate ale unui task pe mașini incompatibile atâta timp cât nu se depășește load-ul maxim