

Universidade Federal do Rio Grande do Sul
Otimização Combinatória
Luciana Salete Buriol
Turma - U

Sequenciamento em Máquinas Paralelas

Alunos: Felipe Mathias Schmidt, Gabriel Gonçalves

Introdução:

O problema de sequenciamento em Máquinas Paralelas consiste em, dado um conjunto de tarefas com tempo de processamento e um conjunto de máquinas, encontrar uma atribuição de tarefas para cada máquina que minimize o término da última operação executada.

Formulação do problema:

Entradas:

- Conjunto de tarefas: $T = [n]$ onde cada tarefa tem um tempo de processamento $p_t, t \in T$.
- Conjunto de máquinas: $M = [m]$

Solução:

- Atribuição de tarefas à máquina: $a: T \rightarrow M$

Objetivo:

- Minimizar o término de execução da última tarefa.

Formulação:

$$\text{Min } C_{\max}$$

$$\text{s.a } \sum_{i \in M} x_{ij} = 1 \quad \forall j \in T$$

$$\sum_{j \in T} p_j x_{ij} \leq C_{\max} \quad \forall i \in M$$

$$x_{ij} \in \{0, 1\}$$

Onde: $T = \text{Tarefas}$

$M = \text{Máquinas}$

$p_j = \text{tempo de processamento da tarefa } j$.

$x_{ij} = 1$; se a tarefa j for executado na máquina i ;

0; caso contrário.

$C_{\max} = \text{makespan (Instante de término da máquina mais carregada)}$

Formulação para GLPK:

Máquinas

param m, integer, > 0;

Tarefas

param t, integer, > 0;

Conjunto de tarefas

set TAREFAS;

Tempo de execução para cada tarefa

param tempos {TAREFAS} >= 0;

Variável de decisão. X_{ij} onde i = Máquinas e j = Tarefas. X_{ij} = Máquina i com Tarefa j
var x {i in 1..m, j in 1..t}, binary;

#Makespan

var z;

Funcao objetivo

minimize Obj: z;

Restrições

Tarefa alocada à exatamente uma máquina

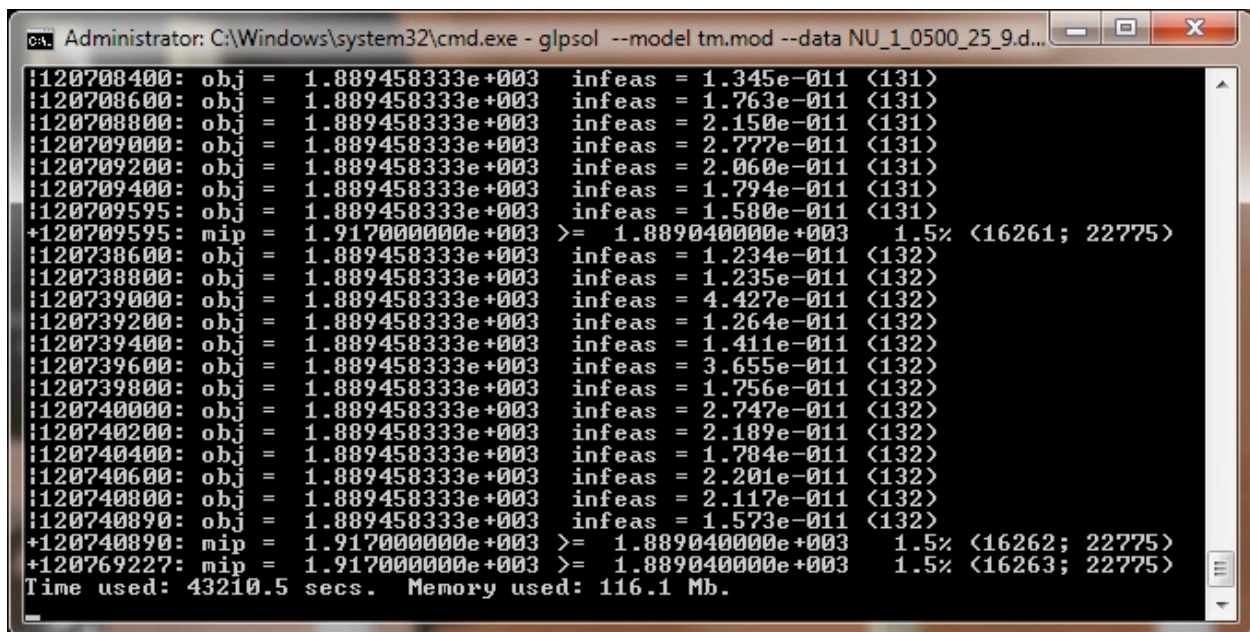
s.t. apenasUmaMaquina{j in 1..t}: sum{i in 1..m} x[i, j] = 1;

Tempo de processamento das máquinas

*s.t. TempoProcMaq{i in 1..m}: z >= sum{j in 1..t} x[i, j] * tempos[j];*

Utilização de Solver:

Usando o GLPK com a formulação dada anteriormente, executamos o solver com os dados de entrada solicitados por um tempo de 15 minutos afim de ver qual seria a resposta por ele informada. Como esperado, este tempo não é suficiente para chegar à resposta ótima. Também deixamos o solver executar durante aproximadamente 12h com a instância de entrada NU_1_0500_9 e da mesma forma não encontramos ainda a solução ótima, mas algo próximo disso, como mostrado na imagem seguinte:



```
Administrator: C:\Windows\system32\cmd.exe - glpsol --model tm.mod --data NU_1_0500_25_9.d...
!120708400: obj = 1.889458333e+003   infeas = 1.345e-011 <131>
!120708600: obj = 1.889458333e+003   infeas = 1.763e-011 <131>
!120708800: obj = 1.889458333e+003   infeas = 2.150e-011 <131>
!120709000: obj = 1.889458333e+003   infeas = 2.777e-011 <131>
!120709200: obj = 1.889458333e+003   infeas = 2.060e-011 <131>
!120709400: obj = 1.889458333e+003   infeas = 1.794e-011 <131>
!120709595: obj = 1.889458333e+003   infeas = 1.580e-011 <131>
+120709595: mip = 1.917000000e+003   >= 1.889040000e+003   1.5% <16261; 22775>
!120738600: obj = 1.889458333e+003   infeas = 1.234e-011 <132>
!120738800: obj = 1.889458333e+003   infeas = 1.235e-011 <132>
!120739000: obj = 1.889458333e+003   infeas = 4.427e-011 <132>
!120739200: obj = 1.889458333e+003   infeas = 1.264e-011 <132>
!120739400: obj = 1.889458333e+003   infeas = 1.411e-011 <132>
!120739600: obj = 1.889458333e+003   infeas = 3.655e-011 <132>
!120739800: obj = 1.889458333e+003   infeas = 1.756e-011 <132>
!120740000: obj = 1.889458333e+003   infeas = 2.747e-011 <132>
!120740200: obj = 1.889458333e+003   infeas = 2.189e-011 <132>
!120740400: obj = 1.889458333e+003   infeas = 1.784e-011 <132>
!120740600: obj = 1.889458333e+003   infeas = 2.201e-011 <132>
!120740800: obj = 1.889458333e+003   infeas = 2.117e-011 <132>
!120740890: obj = 1.889458333e+003   infeas = 1.573e-011 <132>
+120740890: mip = 1.917000000e+003   >= 1.889040000e+003   1.5% <16262; 22775>
+120769227: mip = 1.917000000e+003   >= 1.889040000e+003   1.5% <16263; 22775>
Time used: 43210.5 secs. Memory used: 116.1 Mb.
```

Na imagem podemos ver que o solver executou aproximadamente 12h (43210.5 segundos) e têm como solução corrente 1889 com um total de 116mb de memória usados, a solução ótima do problema é 1879, bem próxima da encontrada até momento.

A máquina utilizada para a execução do solver, tanto para a execução de 12h como as de 15min, tem um processador Intel Core 2 Quad de 2.33Ghz e 4Gb de memória RAM.

Metaheurística

Foi implementada a metaheurística Simmulated Annealing, que, dada uma solução inicial, gera perturbações randômicas, chamadas vizinhos, no intuito de otimizá-la. Um vizinho substitui uma solução inicial sempre que for mais ótima. Senão,

substitui com uma probabilidade que diminui a cada iteração do algoritmo. Utilizamos uma função de probabilidade de aceitação bastante comum,

$$\mu = \exp(-\lambda \times \Delta L/L).$$

Onde $\lambda = (k/\beta)$ (k é o número de iterações realizadas até o momento e β é uma constante experimental, neste caso 300. Testes com constantes maiores revelaram pioras nos resultados), L é o valor da função objetivo (no caso, o tempo até o final da execução do último programa) e ΔL é a diferença entre a função objetivo da solução inicial e do vizinho. Para que o vizinho substitua a solução inicial, a seguinte condição deve ser satisfeita:

$$\mu > r$$

r um número randômico, sendo $0 \leq r \leq 1$.

O algoritmo para após 15 minutos, ou após 30.000 iterações sem melhoras na função objetivo. Para a geração da solução inicial foi usado um algoritmo LPT (Longest Processing Time), atribuindo o processo de maior tempo à máquina mais ociosa. Testes foram feitos também com 3.000.000 iterações, revelando melhoras significativas.

Finalmente, para a geração de vizinhos, troca-se um processo aleatório da máquina com maior tempo de processamento com um processo aleatório das demais máquinas. A máquina utilizada para a execução da metaheurística tem um processador Intel Core 2 duo de 1.60Ghz e 2Gb de memória RAM.

Análise de Resultados:

Após a execução da metaheurística e do solver GLPK pelo tempo de 15 minutos para cada entrada, conseguimos os dados que estão arranjados nas tabelas seguintes:

- Comparação entre heurística (30.000 iterações)/GLPK e Solução Ótima:

Entrada	Solução Ótima	GLPK	Diferença (%)	Heurística	Diferença (%)
NU_1_0500_25_9	1879	1932	2,820649282	1906	1.4369345396
NU_1_1000_25_9	3763	4907	30,40127558	3784	0.5580653734
NU_2_0500_25_9	18784	19404	3,300681431	19140	1.895229983
NU_2_1000_25_9	37623	38226	1,602743003	37974	0.9329399569
NU_3_0500_25_9	187833	193857	3,207104183	190480	1.4092305399
NU_3_1000_25_9	376225	397346	5,613927836	378393	0.5762509137
U_1_0500_25_9	988	1301	31,68016194	992	0.4048582996
U_1_1000_25_9	2048	3180	55,2734375	2062	0.68359375
U_2_0500_25_9	9684	10808	11,60677406	9816	1.3630731103
U_2_1000_25_9	19120	23707	23,99058577	19274	0.8054393305
U_3_0500_25_9	97050	115487	18,99742401	97676	0.6450283359
U_3_1000_25_9	200508	202389	5,613927836	202432	0.9595627107

- Comparação entre heurística (3.000.000 iterações)/GLPK e Solução Ótima:

Entrada	Solução Ótima	GLPK	Diferença (%)	Heurística	Diferença (%)
NU_1_0500_25_9	1879	1932	2,820649282	1890	0.585417775
NU_1_1000_25_9	3763	4907	30,40127558	3772	0.239170874
NU_2_0500_25_9	18784	19404	3,300681431	18888	0.553662692
NU_2_1000_25_9	37623	38226	1,602743003	37716	0.247189219
NU_3_0500_25_9	187833	193857	3,207104183	188870	0.552086162
NU_3_1000_25_9	376225	397346	5,613927836	377229	0.266861585
U_1_0500_25_9	988	1301	31,68016194	988	0
U_1_1000_25_9	2048	3180	55,2734375	2048	0
U_2_0500_25_9	9684	10808	11,60677406	9684	0
U_2_1000_25_9	19120	23707	23,99058577	19123	0.015690377
U_3_0500_25_9	97050	115487	18,99742401	97089	0.040185471
U_3_1000_25_9	200508	202389	5,613927836	200528	0.009974664

- Comparação entre heurística e GLPK:

Comparação solução encontrada			
Entrada	GLPK	Heurística	Melhora Heurística (%)
NU_1_0500_25_9	1932	1890	2.23523150612027
NU_1_1000_25_9	4907	3772	30.1621047036939
NU_2_0500_25_9	19404	18888	2.74701873935264
NU_2_1000_25_9	38226	37716	1.35555378358983
NU_3_0500_25_9	193857	188870	2.65501802132745
NU_3_1000_25_9	397346	377229	5.34706625024918
U_1_0500_25_9	1301	988	31.6801619433198
U_1_1000_25_9	3180	2048	55.2734375
U_2_0500_25_9	10808	9684	11.6067740603057
U_2_1000_25_9	23707	19123	23.9748953974895
U_3_0500_25_9	115487	97089	18.9572385368367
U_3_1000_25_9	202389	200528	0.928142518004279

Para a metaheurística, em todos os casos, o algoritmo parou antes dos 15 minutos de tempo limite, utilizando como critério de parada tanto as 30 mil quanto as 3 milhões iterações sem melhoria de resultado como critério de parada.

Conclusão:

Partindo da análise dos dados obtidos neste trabalho, vemos que a metaheurística apresenta resultados muito superiores ao do GLPK, e em tempo muito menor, principalmente para entradas maiores, onde a diferença percentual entre a solução ótima e a solução estudada são ainda maiores.

Bibliografia:

- <http://www.decom.ufop.br/prof/marcone/Disciplinas/OtimizacaoCombinatoria/OtimizacaoCombinatoria.pdf>
- http://pt.wikipedia.org/wiki/Simulated_annealing
- <http://www.inf.ufrgs.br/~mrpritt/lib/exe/fetch.php?media=inf05010:e06a1.pdf>
- <http://www.ibm.com/developerworks/linux/library/l-glpk2/>
- <http://www.g12.cs.mu.oz.au/mzn/hakank/jssp.mzn>

- <https://wiki.hi.is/index.php/A%C3%B0ger%C3%B0agreining/glpk:Jssp.mod>
- <http://www.huttenhower.org/hg/hgwebdir.cgi/humann/annotate/24827c787832/MinPath/glpk-4.6/examples/jssp.mod>