UMA ABORDAGEM ACO COM RECONEXÃO POR CAMINHOS PARA O PROBLEMA DE ESCALONAMENTO DE TAREFAS EM MÁQUINAS PARALELAS NÃO RELACIONADAS COM PENALIZAÇÃO POR ADIANTAMENTO E ATRASO

Celio H. N. Larcher Jr.
Laboratório Nacional de Computação Científica - LNCC
Av. Getúlio Vargas, n 333, Quitandinha - Petrópolis - RJ
clarcher@lncc.br

Stênio Sã R. F. Soares, Luciana Brugiolo Gonçalves
Universidade Federal de Juiz de Fora - UFJF
Rua José Lourenço Kelmer, s/n, Campus Universitário - Juiz de Fora - MG
ssoares@ice.ufjf.br, lbrugiolo@ice.ufjf.br

RESUMO

Neste trabalho é proposta uma heurística híbrida para tratar o Problema de Sequenciamento de Tarefas em Máquinas Paralelas não Relacionadas, onde são considerados os tempos de preparação dependentes da sequência de tarefas, bem como a possibilidade de inclusão de tempo de ociosidade nas máquinas. O objetivo do problema é a minimização das penalização por antecipação e atraso. O algoritmo proposto combina Otimização por Colônia de Formigas (ACO) e *Path Relinking* (PR), em que se aplica uma técnica distinta de manutenção do conjunto elite. Os resultados indicam que a abordagem é competitiva quando comparados aos obtidos por um trabalho recente da literatura, que propôs um algoritmo baseado em GRASP e PR.

PALAVRAS CHAVE. Escalonamento de Tarefas, Colônia de Formigas, Path Relinking.

Tópicos: Metaheurísticas, Otimização Combinatória, Logística.

ABSTRACT

This paper proposes a hybrid heuristic to deal with the Unrelated Parallel Machine Problem, which are considered the preparation time of jobs as well as the possibility of including idle time on the machines. The objective of the problem is to minimize the total earliness and tardiness penalties. The proposed algorithm combines Ant Colony Optimization (ACO) and Path Relinking (PR), applying a different technique of maintaining the elite set. The results show that the approach is competitive when compared to those obtained in a recent study of literature that proposed an algorithm based on GRASP and PR.

KEYWORDS. Job Scheduling, Ant Colony Optimization, Path Relinking.

Paper topics: Metaheuristics, Combinatorial Optimization, Logistics.

1. Introdução

O escalonamento de tarefas em linhas de produção tem grande importância em diversos processos de manufaturas, desde em linhas tradicionais, como na produção de veículos e eletrodomésticos, como em outras não tão evidentes, como na indústria química e metalúrgica. A importância está, em geral, ligada ao caráter estratégico que estas decisões podem proporcionar, como a diminuição de gastos com multas por atraso na entrega do produto ou mesmo os custos gerados com armazenagem de produtos por conta de antecipação da produção.

Dada a sua importância, diversas variantes do problema de escalonamento de tarefas foram largamente estudadas na literatura ([Kramer e Subramanian, 2015]). Uma destas variações é relacionada ao uso de máquinas paralelas, possuindo grande relevância prática e podendo ser dividido em três tipos principais conforme as máquinas: máquinas paralelas idênticas, quando não há diferenças entre máquinas da linha de produção; máquinas paralelas uniformes, quando há diferença apenas na velocidade de produção das máquinas; e máquinas paralelas não-relacionadas, quando há comportamento específico para cada máquina presente na linha de produção.

O critério utilizado para a definição da ordem de execução das tarefas é, talvez, o fator determinante na definição de problemas de escalonamento de tarefas, sendo os critérios relacionados ao atraso ponderado e atraso máximo na data de entrega os de maiores aplicações práticas. Apesar do extensivo uso destes critérios, outra visão que vem ganhando notoriedade refere-se à aplicações *Just in Time* ([Ohno, 1988]), onde deseja-se que cada tarefa seja concluída o mais próximo possível de sua data de entrega, minimizando custos relacionados com uma entrega muito adiantada, por exemplo custos de estocagem, além dos problemas relacionados a atraso.

O trabalho aqui proposto trata de uma das variações do problema de máquinas paralelas não-relacionadas, que tem como objetivo minimizar o atraso e adiantamento ponderado do tempo de conclusão de cada tarefa, considerando tempo ocioso entre tarefas e a existência de tempo de preparação específico para cada par de tarefas conforme cada máquina.

Esta variação do problema de escalonamento já foi considerada em outros trabalhos da literatura. Em [Vallada e Ruiz, 2012] os autores apresentam um Algoritmo Genético com rotina para inserção dos tempos ociosos. Em [Nogueira et al., 2014] é proposta uma heurística híbrida que combina uma heurística GRASP (*Greedy Randomized Adaptive Search Procedure*) com as técnicas *Iterated Local Search* (ILS) e *Path Relinking* (PR) e que apresentou alguns bons resultados. Em [Kramer e Subramanian, 2015] os autores apresentam a técnica UILS, uma variação do ILS combinado com RVND voltado a resolução de diversos problemas de escalonamento.

Neste trabalho é proposta uma abordagem híbrida que combina Otimização por Colônia de Formigas (ACO) com a técnica PR. Um algoritmo ACO consiste em uma técnica baseada em inteligência coletiva que simula o comportamento de uma colônia de formigas, tendo sido amplamente utilizada em uma gama de variações de problemas de escalonamento ([Arnaout et al., 2008], [Gutjahr e Rauner, 2007]). O procedimento de PR se apresenta como uma técnica de intensificação no processo de exploração do espaço de soluções, e também já foi utilizada em trabalhos que tratam problemas de escalonamento ([Nogueira et al., 2014]).

As demais seções se apresentam da seguinte forma: a Seção 2 apresenta o modelo matemático do problema, a Seção 3 apresenta a abordagem proposta para sua resolução, enquanto a Seção 4 apresenta os resultados obtidos e comparação com a literatura e a Seção 5 apresenta as conclusões do trabalho e proposta de trabalhos futuros.

2. Modelo Matemático

Para uma descrição formal do problema, considere $N = \{1, 2, \ldots, n\}$ o conjunto de tarefas a serem processadas e $M = \{1, 2, \ldots, m\}$ o conjunto das máquinas disponíveis para executar tais tarefas. Para cada tarefa $i \in N$ são conhecidos: data de entrega d_i, penalidade por atraso β_i, penalidade por adiantamento α_i e os tempos de execução em cada máquina $k \in M$, denotado por p_{ik}. Além destas informações, o tempo de preparação (tempo de *setup*) entre duas tarefas consecutivas i e j em uma dada máquina k e dado por s_{ijk}. Não é necessário tempo de preparação antes

da execução da primeira atividade em cada máquina, não é permitida preempção e cada máquina só pode executar uma atividade por vez. Permite-se a inclusão de tempos de ociosidade (*idle time*), ou seja, período de tempo em que a máquina não executa nenhuma tarefa. Considere que todas as atividades estão disponíveis para processamento no instante inicial.

A formulação matemática que segue foi apresentada no trabalho de [Nogueira et al., 2014]. A variável binária x_{ijk} assume valor 1 se a tarefa j é processada após a tarefa i na máquina k, 0 caso contrário. A variável C_{ik} armazena o instante de conclusão da tarefa i na máquina k. Para registrar o adiantamento e atraso das tarefas são utilizadas as varáveis E_i e T_i, respectivamente, onde $E_i = max\,(d_i - C_i, 0)$ e $T_i = max\,(C_i - d_i, 0)$. Segue o modelo de programação linear.

$$\min \sum_{i=1}^{n} (\alpha_i E_i + \beta_i T_i) \tag{1}$$

$$\sum_{k=1}^{m} \sum_{i=0, i\neq j}^{n} x_{ijk} = 1, \qquad \forall j \in N \tag{2}$$

$$\sum_{j=1}^{n} x_{0jk} \leq 1, \qquad \forall k \in M \tag{3}$$

$$\sum_{i=0, i\neq h}^{n} x_{ihk} - \sum_{j=0, j\neq h}^{n} x_{hjk} = 0, \qquad h \in N, \forall k \in M \tag{4}$$

$$C_{0k} = 0, \qquad \forall k \in M \tag{5}$$

$$C_{jk} \geq C_{ik} - \mathbb{M} + (p_{jk} + s_{ijk} + \mathbb{M}) \times x_{ijk}, \qquad \forall i, j \in N, \forall k \in M \tag{6}$$

$$E_i \geq d_i - C_{i,k}, \qquad \forall i \in N, \forall k \in M \tag{7}$$

$$T_i \geq C_{i,k} - d_i, \qquad \forall i \in N, \forall k \in M \tag{8}$$

$$T_i \geq 0, E_i \geq 0, \qquad \forall i \in N \tag{9}$$

$$x_{ijk} \in \{0, 1\}, \qquad \forall i, j \in N, \forall k \in M \tag{10}$$

A função objetivo (1) busca minimizar as penalizações por antecipação e atraso. As restrições (2) garantem que cada tarefa é atribuída a exatamente uma máquina. As restrições (3) e (4) asseguram que cada máquina utilizada tem uma única sequência de tarefas associada, sempre iniciando da tarefa 0. Para as tarefas 0, fictícias, a restrição (5) determinam que o tempo de conclusão é zero. Nas restrições (6) são definidos os tempo de conclusão das demais tarefas, utilizados nas restrições (7) e (8) para cômputo do adiantamento e atraso. As restrições (9) e (10) indicam o domínio das variáveis do modelo.

Devido à natureza combinatória do problema (NP-Difícil), foi proposta uma abordagem heurística para buscar soluções de boa qualidade em tempo adequado para aplicações reais.

3. Abordagem proposta

Este trabalho propõe a utilização de uma heurística híbrida para resolução do problema denominada ACO+PR. Para isto é utilizado um algoritmo construtivo em duas fases: uma implementação da metaheurística ACO e um processo aleatório guloso.

Para aprimoramento da solução é utilizado ainda um algoritmo *Path Relinking*, com constante renovação do conjunto elite, além de uma técnica de busca local baseada na heurística NEH ([Nawaz et al., 1983]).

Para determinação dos tempos ótimos de início de cada tarefa, considerando a existência de tempo ocioso, foi ainda aplicada a rotina apresentada em [Nogueira et al., 2014].

3.1. Representação e avaliação das soluções

Para codificar a solução foi utilizado um vetor de *m* listas encadeadas, cada uma destas listas associada a uma máquina. Em cada lista, as tarefas são apresentadas consecutivamente em ordem de tempo de processamento, com cada tarefa armazenando seu tempo de início.

A avaliação de uma solução é feita pela aplicação direta da Equação 1, considerando o tempo de término de cada tarefa no vetor de listas de tarefas associado a cada máquina.

3.2. Estrutura principal

A estrutura da heurística híbrida, que pode ser vista no (Algoritmo 1), é composta por uma fase de construção e outra de aprimoramento das soluções. A fase de construção consiste em uma implementação da metaheurística ACO para obtenção de boas e diversas soluções para a próxima fase (linhas de 6 a 12), tendo as melhores soluções encontradas passadas ao processo de aprimoramento.

Na fase de aprimoramento, aplicada em intervalos regulares de *Apri_Fase* iterações (linha 17), o *Path Relinking* é executado (linha 18) utilizando as soluções do conjunto elite e gerando novas soluções. Finalmente, a melhor solução encontrada é passada ao processo de busca local (linha 20) de forma a ainda aprimorar a qualidade da solução obtida no *Path Relinking*.

Algorithm 1 ACO+PR

```
1:  procedure ACO+PR(ANT_MAX, α, β, ρ, δ, γ, pressLinear, Apri_Fase, limiarSimilaridade, max_PR)
2:      InicializaFeromonio (Feromônio);
3:      it ← 0;
4:      Elite ← ∅;
5:      while Critério de Parada do
6:          for ant de 1..ANT_MAX do
7:              solução ← Construtivo_ACO (Feromônio, α, β, δ, γ, pressLinear);
8:              Atualização do Feromônio(ρ);
9:              if solução melhor que solBest_It then
10:                 solBest_It ← solução;
11:             end if
12:         end for
13:         if solBest_It melhor que solBest then
14:             solBest ← solBest_It;
15:         end if
16:         Elite ← Elite ⋃ solBest_It;
17:         if it mod Apri_Fase = 0 then
18:             Elite ← PathRelinkingSet (Elite, limiarSimilaridade, max_PR);
19:             solBest_PR ← MelhorSolução (Elite);
20:             solBest_PR_BL ← Busca_Local(solBest_PR);
21:             if solBest_PR_BL melhor que solBest then
22:                 solBest ← solBest_PR_BL;
23:                 Elite ← Elite ⋃ solBest_PR_BL;
24:             end if
25:         end if
26:         it ← it + 1;
27:     end while
28: end procedure
```

3.3. Construtivo ACO

Em problemas de escalonamento em geral, e neste em específico, são nítidas pelo menos duas decisões bem distintas a serem tomadas na construção de um solução: a decisão da ordem das tarefas a serem inseridas e a escolha do local em que cada uma destas será incluída.

A estratégia proposta, Algoritmo 2, trata cada uma destas decisões de maneira distinta. Para escolha da ordem em que cada tarefa será inserida na solução foi utilizada uma implementação

baseada na metaheurística ACO (Fase 1, linhas 3-13), enquanto para determinar a máquina em que cada tarefa será inserida (Fase 2, linhas 14-22) um algoritmo guloso aleatório foi usado.

Algorithm 2 Construção de uma solução

```
1:  function CONSTRUTIVO_ACO(Feromônio, α, β, δ, γ, pressLinear)
2:      /*Fase 1: Definição da ordem de inserção das tarefas*/
3:      vetTarefas[nTarefas];        //Vetor com tarefas em ordem de inserção
4:      LC ← Tarefas em ordem crescente de data de entrega;
5:      for Tarefa i ← 1 ... |LC| do
6:          Determinar η_i (pressLinear); /* Fórmula do Ranking Linear (Equação 11)*/
7:      end for
8:      for i de 1 ... nTarefa do
9:          LRC ← (δ× nTarefas) primeiras tarefas de LC;
10:         tarefaEscolhida ← RoletaTarefa(LRC, Feromonio, α, β);
11:         vetTarefas[i] ← tarefaEscolhida;
12:         LC ← LC \ {tarefaEscolhida};
13:     end for
14:     /*Fase 2: Atribuição das tarefas às máquinas*/
15:     for i de 1 ... nTarefa do
16:         for j de 1 ... mMáquinas do
17:             Determinar Δ_ij; /* impacto de inserção de vetTarefas[i] ao fim da máquina j*/
18:         end for
19:         LRC_Maq ← (γ× mMaquinas) máquinas com menor Δ_ij;
20:         máquinaEscolhida ← RoletaMáquina (LRC_Maq);
21:         InserirTarefa(solução, vetTarefas[i], máquinaEscolhida);
22:     end for
23:     return solução;
24: end function
```

Fase 1: Definição da Ordem de Inserção das Tarefas

Na implementação do ACO, a data de entrega de cada tarefa é utilizada como informação heurística. Esta informação é disposta num *ranking* linear em ordem crescente. A contribuição heurística η_i associada a uma tarefa *i* (linhas 5-7 do Algoritmo 2) é definida pela Equação 11,

$$\eta_i = 2 - L + 2 * \frac{(L-1)(n - P_i + 1)}{n-1} \tag{11}$$

onde L é o parâmetro de pressão linear, P_i é a posição na ordenação por data de entrega de uma tarefa i e n é o número total de tarefas.

Para determinar a parcela responsável pelo aprendizado (feromônio) foi definida uma relação entre as tarefas e sua posição no vetor de inserção. Nesta abordagem, a cada k posições do vetor é definido um valor para o feromônio, sendo $k = n/m$ onde m é o número de máquinas. Ou seja, são gerados m blocos com k inserções, cada bloco b representado por I_b, como mostra a matriz de feromônios abaixo.

$$\begin{array}{c} \\ T_1 \\ T_2 \\ \vdots \\ T_n \end{array} \begin{array}{c} \begin{array}{cccc} I_0 & I_2 & \cdots & I_{m-1} \end{array} \\ \begin{bmatrix} \tau_{1,0} & \tau_{1,1} & \cdots & \tau_{1,m-1} \\ \tau_{2,0} & \tau_{2,1} & \cdots & \tau_{2,m-1} \\ \vdots & \vdots & \vdots & \cdots \\ \tau_{n,0} & \tau_{n,1} & \cdots & \tau_{n,m-1} \end{bmatrix} \end{array}$$

Desta forma, para determinar a ordem em que as tarefas serão inseridas na solução (preenchimento do vetor vetTarefas[ ], linhas 8-13), a partir da Lista de Candidatos - LC contendo todas as tarefas em ordem crescente pela data de entrega, são selecionadas as $\delta\times$ *nTarefas* primeiras tarefas de LC para comporem a Lista Restrita de Candidatos, LRC. Para determinar, entre as tarefas de LRC, aquela que será inserida no vetor vetTarefas, é utilizado o procedimento RoletaTarefa (linha 10), que determina, através da fórmula comum à implementações de ACO, a probabilidade P_i de escolha de cada tarefa *i* de LRC, conforme a Equação 12,

$$P_i = \frac{\tau_{ib}^{\alpha}\eta_i^{\beta}}{\sum_i \tau_{ib}^{\alpha}\eta_i^{\beta}} \tag{12}$$

sendo b o bloco referente ao momento da inserção, α e β os parâmetros para ajuste de influência de cada tipo de informação, feromônio e heurística, respectivamente. Assim, nas inserções relacionados ao bloco b, da $(b \times k + 1)$ a $(b + 1) \times k$ ésima inserções, são considerados os feromônios da coluna I_b.

Fase 2: Atribuição das Tarefas às Máquinas

Dada a ordem de inserção das tarefas, definida na Fase 1, na segunda fase estas tarefas serão atribuídas, uma a uma, às máquinas (linhas 15-22). Para a inserção da i-ésima tarefa é verificado o impacto, em cada máquina j, se esta for incluída após a última tarefa alocada (Δ_{ij}). Considerando esta informação, uma Lista Restrita (LRC_Maq) com as máquinas com menor valor de Δ_{ij} é construída (linha 19).

Para determinar a máquina em que a i-ésima tarefa será inserida é feito um procedimento baseado no método da roleta. Neste procedimento, as máquinas $j \in$ LRC_Maq têm probabilidade de escolha proporcional $1/\Delta_{ij}$.

Este processo não leva em consideração informações sobre o tempo ocioso. A probabilidade de escolha de uma máquina k é apresentada na Equação 13.

$$P_k = \frac{1}{|C_{max+i}^{k} - D_i| * penalidade + 1} \tag{13}$$

onde P_k é a probabilidade de seleção da máquina k, C_{max+i}^{k} é o tempo final esperado ao se inserir a tarefa i na máquina k, D_i é o instante de entrega esperado para a tarefa i, definido na instância, e $penalidade$ é o valor de penalidade aplicado pelo não cumprimento do prazo de entrega, também definido na instância (α se adiantamento, ou β se atraso).

3.4. Estratégia de Atualização do Feromônio

A etapa de construção descrita anteriormente está inserida em uma estrutura tradicional de implementações para um algoritmo de Otimização por Colônia de Formigas, onde cada solução está relacionada ao caminho percorrido por uma formiga. Após a construção, cada formiga, de acordo com a qualidade da solução gerada, altera o valor do feromônio referente a cada uma das escolhas que esta realizou.

Para a atualização do valor do feromônio, conforme apresentado na Equação 14, é feita uma relação entre o valor da função objetivo da melhor solução já encontrada (*bestSolution*) e o valor da solução corrente (*scoreSolution*).

$$\tau_{ib} = \tau_{ib}(1 - \rho) + \frac{bestScore}{scoreSolution} \times \rho \tag{14}$$

com τ_{ib} sendo o valor do feromônio para a tarefa i, inserida no bloco b, e ρ o parâmetro de evaporação. Isto permite o balanceamento entre o valor encontrado no processo heurístico e de aprendizado, mantendo-os na mesma ordem de grandeza. O valor para todos os feromônios é limitado no intervalo (0,2), de forma a não haver grande discrepância entre os valores e permitir uma melhor exploração do espaço de busca.

3.5. O Algoritmo *Path Relinking*

A técnica *Path Relinking* [Glover, 1996] é uma heurística de aprimoramento de soluções que busca explorar o espaço entre duas soluções, criando um caminho de conexão entre estas. O processo faz uso de duas soluções (uma chamada solução de origem e a outra, solução guia) e realiza movimentos sucessivas que tornam a solução origem cada vez mais similar a solução alvo, guiando-se sempre pelos movimentos que mais aprimoram a qualidade da solução. O processo termina quando as soluções são idênticas.

Na implementação realizada, os movimentos ocorrem de duas formas: através de inserções de tarefas ao fim de uma máquina, ou através da troca entre duas tarefas. Em cada passo, são verificadas todas as trocas e inserções que possam colocar ao menos uma tarefa em sua posição correta. A melhoria proporcionada por cada mudança é estimada (mas sem a determinação exata do tempo ocioso ótimo). Aquele movimento que levar a uma melhor solução é realizado, mesmo que leve a uma solução de pior qualidade comparada à original.

Neste trabalho, optou-se pela versão conhecida como *Path Relinking* Misto, onde as soluções origem e guia alternam o seu papel. Após a escolha do movimento realizado em cada etapa, a solução resultante têm seu tempo ocioso ótimo determinado através do procedimento descrito na Seção 3.7. O *Path Relinking* retorna a melhor solução encontrada durante todo o processo. Um exemplo pode ser visto na Figura 1.

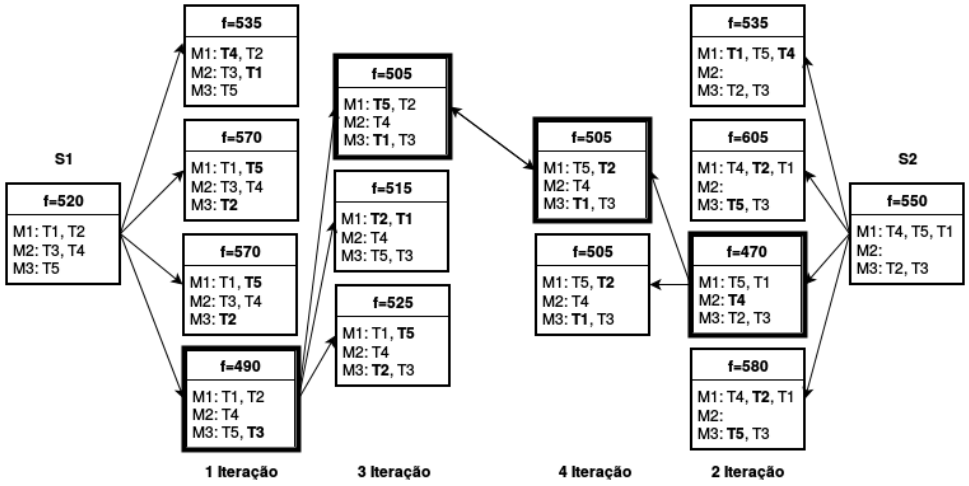

Figura 1: Exemplo *Path Relinking* Misto

No exemplo, dadas duas soluções (S1 e S2), inicia-se o processo aproximando a configuração de S1 em relação a de S2. Para isto, analisam-se todos os possíveis movimentos individuais que tragam alguma tarefa de S1 para a posição em que esta está presente em S2 (repare que a tarefa T1 não pode ser posta na posição correta com apenas um movimento). Após essa analise, a solução S1 é alterada para a melhor dentre as possibilidades analisadas e repete-se o processo, agora verificando as mudanças possíveis na solução S2 que aproximem esta da nova Solução S1. O processo se repete até que as soluções sejam idênticas.

3.6. Abordagem de Aprimoramento por *Path Relinking*

Para realização do *Path Relinking*, um conjunto de soluções denotado Elite é mantido, onde são inseridas as soluções que serão submetidas ao processo de aprimoramento. Para compor o conjunto Elite, a cada iteração da fase de construção (Algoritmo 1 - linhas de 6-12), a melhor solução é inserida. Além destas, são inseridas aquelas encontradas após o processo de *Path Relinking* e Busca Local, caso esses resultem em melhoria da solução.

O Algoritmo 3 apresenta o funcionamento do processo de aprimoramento usado neste trabalho. Durante o processo, o subconjunto *Selected_Set* é extraído do conjunto Elite de tal forma

que contenha as max_PR melhores e mais distintas soluções (linhas 3-12). Para isso, as soluções são ordenadas de acordo com valor da função objetivo.

A partir desta ordenação, as soluções são avaliadas sucessivamente segundo uma função de similaridade em relação a todas as soluções já inseridas em *Selected_Set*. Caso não haja nenhuma outra solução que seja mais similar que um dado limiar, a solução passa a integrar o conjunto. A similaridade entre duas soluções é definida pelo número de coincidências de uma mesma tarefa presente na mesma máquina e com mesmo número de tarefas a precedendo nas duas soluções.

Uma vez construído o conjunto *Selected_Set*, todos os pares de soluções deste conjunto são submetidos ao processo de *Path Relinking*. Ao final deste processo, o conjunto Elite é atualizado de forma a conter apenas as soluções geradas durante este procedimento (linhas 14-16).

Algorithm 3 Aprimoramento utilizando Path Relinking

```
1:  function PATH RELINKING SET(Elite, limiarSimilaridade, max_PR)
2:      Selected_Set ← ∅;
3:      while Elite ≠ ∅ do
4:          solução ← arg min_{s∈Elite} custo(s);
5:          if Similaridade(solução, Selected_Set) < limiarSimilaridade then
6:              Selected_Set ← Selected_Set ⋃ {solução};
7:          end if
8:          Elite ← Elite \ {solução};
9:          if |Selected_Set| = max_PR then
10:             break;
11:         end if
12:     end while
13:     Elite ← ∅;
14:     for ∀ {sol_1, sol_2} ⊆ Select_Set do
15:         Elite ← Elite ⋃ PathRelinking(sol_1, sol_2);
16:     end for
17:     return Elite;
18: end function
```

3.7. Definição dos tempos ociosos ótimos

Para definição dos tempos ociosos ótimos de cada tarefa é usada uma abordagem proposta em [Szwarc e Mukhopadhyay, 1995] e [Wan e Yen, 2002], para o caso de uma única máquina, e ajustado para máquinas paralelas por Nogueira et al. [2014]. Este método busca encontrar os melhores tempos de início de cada tarefa de forma a minimizar a penalidade por atraso e adiantamento.

Na execução do algoritmo, para cada máquina, a partir da primeira tarefa alocada, uma dada tarefa é posicionada de forma que sua data de término coincida com a data esperada para entrega ou o mais próximo possível desta. Se a tarefa corrente não puder ser posta de forma a coincidir com sua data de entrega e existam tarefas anteriores, esta e todas as tarefas que a precedem e que não possuam tempo ocioso entre si são consideradas como um único bloco.

O algoritmo verifica os tempos possíveis de início para o bloco de forma que, para cada tarefa, seu tempo de término coincida com a data de entrega esperada ou o mais próximo possível deste. Para cada uma destas operações é avaliado o impacto na qualidade da solução e a melhor operação é realizada. Note que a escolha de uma posição do bloco pode fazer com que a data de entrega da tarefa corrente não seja atendida. Caso isto aconteça e existam tarefas precedendo o bloco, todas as tarefas sem tempo ocioso entre si, que são anteriores a este bloco, passam a ser consideradas partes do bloco, com o processo refeito recursivamente. A execução ocorre individualmente para cada máquina e com o procedimento sendo finalizado quando todas as tarefas são postas em seu tempo de início ótimo para a solução.

3.8. Busca Local

A busca local implementada neste trabalho, também vista em [Nogueira et al., 2014], baseia-se na conhecida heurística construtiva NEH [Nawaz et al., 1983]. Este método gera uma estrutura de vizinhança através de reinserções de tarefas em todas as posições possíveis e escolhe o melhor entre este conjunto de movimentos.

Mais precisamente, uma tarefa é removida de sua posição original e inserida em todas as outras posições possíveis em todas as máquinas com a qualidade de cada uma destas novas soluções avaliada. O processo se repete para todas as tarefas, e a solução proveniente do movimento de maior ganho substitui a solução original, ponto onde a busca se repete para esta nova solução. Caso não haja movimento que resulte em melhora, o processo é finalizado. Nesta busca local, cada um dos movimentos é avaliado considerando o procedimento que determina os tempos ociosos ótimos.

4. Resultados

As abordagens apresentadas neste trabalho foram testadas para um conjunto de instâncias disponibilizadas por Nogueira [2011]. O pacote é composto por um conjunto de instâncias ditas pequenas (3 e 5 máquinas com 8, 9 e 10 tarefas), médias (3 e 5 máquinas com 20 e 30 tarefas) e instâncias grandes (10, 15, 20, 25 e 30 máquinas com 50, 60, 70, 80, 90 e 100 tarefas). Para cada combinação de número de máquinas com número de tarefas tem-se 10 instâncias.

Os testes foram realizados em uma máquina Intel Core i5 4210U 1.7 GHz, com 6 GB de memória RAM e sistema *OpenSuse* Linux 13.2, com abordagem implementada em linguagem C++ utilizando uma única *thread*.

Para definição dos parâmetros dos algoritmos utilizou-se a ferramenta *irace* [López-Ibáñez et al., 2011], que, dado um grupo de parâmetros com seus intervalos, busca as combinações de valores que apresentam um melhor desempenho nas instâncias fornecidas no teste. O critério de parada utilizado segue o apresentado na literatura para problemas deste tipo, com (n*m*50) milissegundos de processamento para cada instância.

Neste contexto foram definidos como parâmetros para implementação do ACO os valores de ANT_MAX=15, $\alpha = 2.15$ e $\beta = 4.7$, $\rho = 0.4$, pressão linear de seleção *pressLinear*=1.10, dimensão da LRC nas tarefas e máquinas como δ=0.5 e γ=0.4, respectivamente. Para fase de aprimoramento com *Path Relinking* foi utilizado um máximo de *max_PR*=50 soluções por execução do procedimento, limiar de similaridade entre soluções *limiarSimilaridade*=70% e frequência de execução de *Apri_Fase*=130 iterações.

O resultado obtido pela abordagem proposta neste trabalho foi comparado com o resultado do *GRASP+ILS+PR*, melhor entre os algoritmos apresentados por Nogueira [2011]. A configuração do hardware utilizado pelo *GRASP+ILS+PR* (descrito em [Nogueira, 2011]), apesar de diferente da utilizada aqui, não possui grande variação no que se refere a capacidade de processamento, não desvirtuando a análise.

A seguir são apresentados, conforme o tamanho das instâncias, os resultados obtidos a partir de cinco execuções independentes por ambas as abordagens, comparados segundo o valor de RPD (*Relative Percentage Deviation*), calculado como visto em [Nogueira et al., 2014]. Os valores em negrito indicam os melhores resultados.

4.1. Instâncias Pequenas e Médias

Para instâncias de pequeno e médio porte, os resultados, agrupado por dimensão da instância, podem ser vistos nas Tabelas 1 e 2, onde a coluna *M* indica o número de máquinas enquanto a coluna *N* mostra o número de tarefas. A referência para o cálculo do RPD nas instâncias de pequeno porte é o valor ótimo obtido via modelo matemático.

Os resultados demonstram um melhor desempenho da heurística proposta, *ACO+PR*, quando comparada à abordagem da literatura *GRASP+ILS+PR*. Pode-se observar que foi possível obter valores próximos ou iguais ao ótimo em todas as instâncias, com redução do RPD em todos os grupos de instâncias onde a abordagem da literatura não obteve a solução ótima.

M	N	ACO+PR	GRASP+PR+ILS
3	8	**0.00**	**0.00**
	9	**0.00**	0.35
	10	**0.00**	**0.00**
5	8	**0.00**	**0.00**
	9[a]	**0.00**	0.01
	10	**0.11**	0.83
Média		**0.02**	0.20

Tabela 1: Análise por RPD: Instâncias de pequeno porte

M	N	ACO+PR	GRASP+PR+ILS
3	20	**0.01**	1.13
	30	**0.06**	2.41
5	20	**0.22**	0.59
	30	**0.10**	0.74
Média		**0.10**	1.22

Tabela 2: Análise por RPD: Instâncias de médio porte

[a]Duas instâncias deste conjunto foram excluídas da análise (instâncias de ID 0 e 1) por apresentarem na literatura um valor inferior ao encontrado via modelo matemático.

4.2. Instâncias Grandes

Para as instâncias consideradas grandes, a Tabela 3 apresenta os resultados. Observa-se que, para todos os grupos de instâncias, o RPD apresentado pela abordagem proposta é inferior ao da abordagem da literatura em pelo menos uma ordem de grandeza. Isto demonstra a eficácia da abordagem que combina Otimização por Colônia de Formigas e *Path Relinking* para o problema.

Tabela 3: Análise por RPD: Instâncias de grande porte

M	N	ACO+PR	GRASP+PR+ILS	M	N	ACO+PR	GRASP+PR+ILS
10	50	**0.49**	8.93	20	80	**2.06**	13.25
	60	**0.68**	9.38		90	**1.44**	12.59
	70	**0.62**	8.23		100	**1.51**	11.87
	80	**0.65**	10.04	25	50	**4.19**	14.73
	90	**0.86**	10.69		60	**2.60**	15.12
	100	**0.81**	7.81		70	**1.70**	14.64
15	50	**1.37**	13.76		80	**2.38**	14.38
	60	**1.41**	11.29		90	**2.15**	12.77
	70	**0.91**	10.33		100	**1.54**	11.62
	80	**0.78**	10.34	30	50	**2.74**	16.43
	90	**1.43**	12.60		60	**4.27**	15.92
	100	**1.04**	11.00		70	**2.72**	15.25
20	50	**1.59**	13.96		80	**1.83**	15.10
	60	**2.18**	15.82		90	**2.27**	12.57
	70	**1.57**	12.69		100	**2.53**	13.98
Média ACO+PR			**1.74**	**Média GRASP+PR+ILS**			12.57

Este comportamento pode ser visto tanto na média final com 1.74% do *ACO+PR versus* 12.57% do *GRASP+ILS+PR*, quanto na Tabela 4, onde observa-se que, para todas as instâncias deste conjunto, o algoritmo *ACO+PR* melhorou o valor mínimo obtido por *GRASP+ILS+PR*.

Como forma de corroborar as observações anteriores, foi feito um teste de análise de variância utilizando o valor do RPD obtido nas instâncias de grande porte. Para isso, utilizou-se a técnica ANOVA, tendo obtido p-valor de 2.2×10^{-16}, o que caracteriza a existência de diferença estatisticamente significativa entre as técnicas.

5. Conclusão

O trabalho aqui proposto apresenta uma nova abordagem para resolução do Problema de Máquinas Paralelas não Relacionadas com Penalização por Antecipação e Atraso, considerando tempo de preparação e tempo ocioso entre tarefas. Combinando as heurísticas ACO e PR, e utilizando um método específico de manutenção do conjunto Elite, foi possível obter resultados expressivos, tanto relacionados ao desempenho da abordagem quanto a sua estabilidade.

Estes resultados tornaram-se ainda mais aparentes quando comparados a outro trabalho existente na literatura, onde se verificou grande ganho, especialmente nas instâncias de maior dimensão, e onde foi possível, para boa parte das instâncias analisadas, melhorar o valor mínimo conhecido.

Como possibilidades de continuação deste trabalho estão o uso de outras técnicas de construção de solução, verificação de outras estruturas de manutenção do conjunto Elite e combinação de diferentes buscas locais para o melhoria da solução.

Agradecimento

Os autores agradecem a FAPEMIG e a FAPERJ pelo financiamento parcial deste trabalho.

Referências

Arnaout, J. P., Musa, R., e Rabadi, G. (2008). Ant colony optimization algorithm to parallel machine scheduling problem with setups. In *2008 IEEE International Conference on Automation Science and Engineering*, p. 578–582.

Glover, F. (1996). Tabu search and adaptive memory programing – advances, applications and challenges. In *Interfaces in Computer Science and Operations Research*, p. 1–75. Kluwer.

Gutjahr, W. J. e Rauner, M. S. (2007). An aco algorithm for a dynamic regional nurse-scheduling problem in austria. *Computers & Operations Research*, 34(3):642 – 666. ISSN 0305-0548. Logistics of Health Care ManagementPart Special Issue: Logistics of Health Care Management.

Kramer, A. e Subramanian, A. (2015). A unified heuristic and an annotated bibliography for a large class of earliness-tardiness scheduling problems. *CoRR*, abs/1509.02384.

López-Ibáñez, M., Dubois-Lacoste, J., Stützle, T., e Birattari, M. (2011). The irace package, iterated race for automatic algorithm configuration. Technical Report TR/IRIDIA/2011-004, IRIDIA, Université Libre de Bruxelles, Belgium.

Nawaz, M., Enscore, E. E., e Ham, I. (1983). A heuristic algorithm for the m-machine, n-job flow-shop sequencing problem. *Omega*, 11(1):91 – 95. ISSN 0305-0483.

Nogueira, J. a. P. C. M. (2011). *Heurísticas Híbridas para o problema de Programação de Tarefas em Máquinas Paralelas Não Relacionadas com Penalidades por Antecipação e Atraso*. PhD thesis, Universidade Federal de Viçosa.

Nogueira, J. a. P. C. M., Arroyo, J. E. C., Villadiego, H. M. M., e Gonçalves, L. B. (2014). Hybrid grasp heuristics to solve an unrelated parallel machine scheduling problem with earliness and tardiness penalties. *Electron. Notes Theor. Comput. Sci.*, 302:53–72. ISSN 1571-0661.

Ohno, T. (1988). *Toyota Production System: Beyond Large-Scale Production*. Productivity Press., Cambridge, MA. ISBN 0-915299-14-3.

Szwarc, W. e Mukhopadhyay, S. K. (1995). Optimal timing schedules in earliness-tardiness single machine sequencing. *Naval Research Logistics (NRL)*, 42(7):1109–1114. ISSN 1520-6750.

Vallada, E. e Ruiz, R. (2012). *Just-in-Time Systems*, chapter Scheduling Unrelated Parallel Machines with Sequence Dependent Setup Times and Weighted Earliness–Tardiness Minimization, p. 67–90. Springer New York, New York, NY. ISBN 978-1-4614-1123-9.

Wan, G. e Yen, B. P.-C. (2002). Tabu search for single machine scheduling with distinct due windows and weighted earliness/tardiness penalties. *European Journal of Operational Research*, 142(2):271 – 281. ISSN 0377-2217.

Tabela 4: Valores de média e mínimo por instância de médio e grande porte

M	N	I	Média (Heur.)	Min (Heur.)	Min (Lit.)
3	20	0	42769	**42769**	43109
		1	55907	**55907**	**55907**
		2	57504	**57504**	**57504**
		3	84169	**84169**	84796
		4	52538.2	**52515**	**52515**
		5	45765	**45765**	47162
		6	55364.4	**55342**	55454
		7	42488.4	**42476**	42538
		8	45643	**45643**	45673
		9	56114	**56114**	**56114**
	30	0	140847	**140571**	142696
		1	107314.6	**107256**	107703
		2	131846.4	**131768**	131956
		3	123006.8	**122913**	125310
		4	101656.8	**101515**	103915
		5	124498.2	**124494**	125221
		6	100786	**100786**	102503
		7	140490	**140490**	144758
		8	117732.8	**117705**	119779
		9	97556	**97556**	98171
5	20	0	36892.6	**36737**	36757
		1	27825.4	**27795**	27948
		2	40048	**40048**	40081
		3	36835	**36835**	36944
		4	39919.8	**39860**	40177
		5	30733	**30733**	**30733**
		6	26589.2	**26507**	**26507**
		7	25951	**25951**	25982
		8	35205	**35205**	35336
		9	29304	**28957**	28966
	30	0	46427.6	**46357**	47066
		1	91595.2	**91580**	96112
		2	87026	**86851**	89971
		3	77298.6	**77209**	77917
		4	62216.8	**62059**	62642
		5	75749	**75749**	77086
		6	63259	**63227**	64767
		7	58141	**58141**	61693
		8	68734.4	**68671**	69357
		9	92879.8	**92817**	93703
10	50	0	72469	**72300**	75780
		1	77090.2	**76917**	78880
		2	96225.8	**95649**	100951
		3	72192.2	**71911**	75594
		4	89249.4	**88883**	93072
		5	60536.2	**59867**	67135
		6	69626.4	**69319**	72181
		7	73277.8	**72880**	79873
		8	74525.2	**74045**	77801
		9	91197	**90971**	95911
	60	0	133879	**133691**	143111
		1	135363	**134489**	143420
		2	97748.8	**96879**	103688
		3	104429	**104165**	108949
		4	97244	**96942**	99283
		5	104225.4	**103206**	112194
		6	96782.6	**95716**	105531
		7	131040.2	**129966**	133836
		8	127144.6	**126364**	135490
		9	121064	**119858**	128767
	70	0	143328.4	**142866**	147563
		1	139574	**138498**	148091
		2	186055.4	**184671**	198591
		3	152423.4	**151730**	161536
		4	129505.4	**128109**	140898
		5	178725.8	**177912**	188431
		6	152796.6	**151955**	160325
		7	144639	**143450**	152742
		8	144084.4	**143163**	148845
		9	172895.2	**172287**	178741
	80	0	239022	**236297**	253440
		1	191272.8	**190303**	199479
		2	170987	**169951**	181293
		3	224258.2	**223511**	241492
		4	179480.4	**178516**	192582
		5	193684	**192536**	207410
		6	177793	**176291**	192418
		7	237953.8	**236507**	252771
		8	160295.6	**158718**	172603
		9	202862.2	**202271**	211926
	90	0	232208.2	**229464**	243457
		1	257846.4	**256060**	273385
		2	262877.6	**259586**	282041
		3	223761.4	**223097**	239808
		4	256555.8	**254659**	285304
		5	212659	**210769**	228872
		6	193308.8	**191710**	209910
		7	281439	**280574**	298401
		8	238897.2	**237180**	258924
		9	286075.8	**281441**	305827
	100	0	329225.2	**327933**	344354
		1	293145.4	**289505**	306697
		2	295505.4	**294192**	306948
		3	382099.6	**378102**	400341
		4	332070.8	**329914**	353973
		5	361991.8	**359467**	372293
		6	337418.6	**334614**	350134
		7	277805	**275818**	301048
		8	287239.4	**283615**	300062
		9	328380	**326035**	342220
15	50	0	48075.8	**47558**	52924
		1	30582.8	**30379**	32768
		2	34180.4	**33769**	38395
		3	32471.4	**32162**	35181
		4	38393.4	**37979**	43549
		5	45757	**45118**	48957
		6	44104.4	**43498**	48520
		7	23440.6	**22704**	25309
		8	57414.6	**56427**	61247
		9	46483.8	**46072**	51903
	60	0	54065.2	**53335**	58038
		1	66401	**65793**	72362
		2	70722.2	**69653**	76045
		3	60310.2	**59869**	64412
		4	69767.6	**69399**	75589
		5	75095.8	**73556**	81149
		6	53436.8	**52476**	56885
		7	81764	**80674**	85924
		8	54933	**54524**	59000
		9	64855.2	**62976**	67310

M	N	I	Média (Heur.)	Min (Heur.)	Min (Lit.)
15	70	0	94008.4	**93044**	101736
		1	87617.8	**86322**	94010
		2	94932	**94556**	102869
		3	108437.4	**106598**	117372
		4	100618.6	**100232**	107981
		5	115568.8	**115090**	122495
		6	104448	**103309**	111818
		7	106744	**105804**	113634
		8	111249.4	**110844**	122036
		9	101296.4	**99990**	107301
	80	0	112595	**111623**	122529
		1	137452.8	**136798**	148123
		2	108642.4	**106251**	114903
		3	131765.6	**131251**	140197
		4	139666.6	**137984**	150264
		5	121395	**120610**	130719
		6	168200.4	**167436**	176060
		7	133119.8	**132133**	143527
		8	135246.2	**134512**	144004
		9	131320.8	**131060**	139706
	90	0	181977.2	**179583**	194765
		1	198353.8	**197594**	214022
		2	158675.2	**156933**	175812
		3	166137.2	**165201**	179378
		4	145164.2	**143070**	153120
		5	145003.4	**142682**	159666
		6	152273.4	**149249**	168207
		7	152912.8	**148364**	167416
		8	134636.6	**132722**	145935
		9	152327.8	**150334**	166950
	100	0	219555.8	**217251**	235692
		1	210696.2	**208678**	223989
		2	227371.6	**224111**	247896
		3	196304	**194903**	211581
		4	190559.2	**188023**	210383
		5	229835.4	**227328**	248447
		6	232058.2	**229108**	246739
		7	200260	**198185**	216476
		8	216722.8	**214981**	235901
		9	207042.2	**205902**	223487
20	50	0	29329.8	**28838**	30676
		1	28216.4	**27692**	31354
		2	27648.6	**27230**	30244
		3	21035.8	**20572**	24103
		4	37946.6	**37467**	40768
		5	30275.2	**29936**	33151
		6	22384.2	**22063**	23836
		7	22997.2	**22546**	23339
		8	39885.4	**39445**	42988
		9	24998.8	**24621**	26511
	60	0	50697.6	**50086**	53850
		1	35897.6	**35337**	40046
		2	48898.8	**47535**	54782
		3	50573.4	**49138**	55652
		4	40156.6	**39377**	43348
		5	44824	**44053**	50894
		6	35765.6	**34945**	38947
		7	42096	**41531**	46048
		8	42376.2	**41391**	47207
		9	39078	**37794**	44535
	70	0	75035.4	**74750**	80936
		1	60960.8	**60541**	65818
		2	68953.6	**67701**	77167
		3	54684.6	**54047**	58307
		4	68971	**66739**	73832
		5	78135.6	**77582**	82922
		6	65899.8	**65075**	70186
		7	62414	**61097**	65150
		8	64026.2	**62239**	68490
		9	64899.4	**64112**	71240
	80	0	91308.6	**89356**	96862
		1	84184	**82145**	90718
		2	104131.4	**102649**	110851
		3	90991.2	**89288**	97170
		4	79417.6	**77328**	87552
		5	67058.6	**65333**	74266
		6	79798	**78325**	84323
		7	77877.8	**76177**	82060
		8	75350.6	**74546**	82885
		9	77630.6	**76039**	82796
	90	0	100946.6	**99436**	112314
		1	123071.2	**121086**	126409
		2	96449.4	**94884**	101715
		3	86063.8	**84296**	93879
		4	123203.4	**120904**	131026
		5	138542.6	**137609**	151728
		6	102370.4	**100817**	110833
		7	108228	**106951**	120508
		8	117314.6	**116461**	125571
		9	84672.6	**83470**	90351
	100	0	124029.2	**120112**	138311
		1	118302	**116720**	129025
		2	118771.2	**117377**	131066
		3	121880.6	**120459**	126316
		4	135062.6	**132838**	145306
		5	152099.2	**149386**	160892
		6	160698	**158658**	174093
		7	139211.4	**137107**	147605
		8	120352.2	**119172**	127267
		9	153569.8	**152333**	163392
25	50	0	16894.6	**16164**	18380
		1	11985.2	**11706**	12790
		2	10196.6	**9308**	10896
		3	20050.2	**19726**	20965
		4	15090.2	**14363**	14991
		5	11689.8	**11091**	12349
		6	19657.8	**19317**	21660
		7	17400.8	**16481**	17837
		8	18132.6	**17531**	19499
		9	15537.8	**15149**	15702
	60	0	33429.4	**32585**	36474
		1	20564.6	**19594**	21434
		2	33424.8	**32383**	35885
		3	38106.8	**37517**	41822
		4	33191.8	**32402**	36890
		5	23963.4	**23483**	26477
		6	34912.6	**33355**	36208
		7	35786	**35339**	38566
		8	28692.2	**28208**	31290
		9	34271.6	**33761**	39897

M	N	I	Média (Heur.)	Min (Heur.)	Min (Lit.)
25	70	0	54423.4	**52870**	59467
		1	48212.6	**47374**	51548
		2	62916.4	**61920**	69657
		3	70721.4	**69634**	76786
		4	40696.2	**39913**	45368
		5	39756.2	**39296**	43149
		6	45323.4	**44946**	49002
		7	43272.8	**42700**	48529
		8	40852	**40143**	45261
		9	45102.2	**44206**	49149
	80	0	57899.6	**56687**	62291
		1	70541.2	**68725**	75829
		2	83970.2	**82743**	92692
		3	42432.8	**42040**	46612
		4	57086.2	**56723**	60778
		5	46073.4	**43684**	51591
		6	59686.6	**58315**	64301
		7	54888.6	**53151**	57605
		8	42794.2	**42230**	46237
		9	58191.4	**56205**	64448
	90	0	84448.8	**82948**	91361
		1	95658.4	**94009**	102468
		2	94989.4	**93946**	101013
		3	93294	**88334**	98673
		4	86728	**85552**	92836
		5	80013.8	**78637**	87894
		6	82380.4	**80714**	90362
		7	80156.2	**77798**	85236
		8	93318.2	**91723**	100566
		9	94301.2	**93164**	103196
	100	0	106623	**104766**	114499
		1	92054.4	**90932**	97812
		2	127845.8	**125410**	136461
		3	108327.2	**106240**	119345
		4	93371.8	**92429**	103955
		5	99759	**98217**	108558
		6	112438.2	**110376**	119645
		7	108848.6	**107115**	116775
		8	96591.4	**95451**	104111
		9	103697.8	**102503**	111415
30	50	0	20851	**20374**	21159
		1	9706.6	**9598**	10996
		2	13107.6	**12474**	14936
		3	15773.6	**15533**	16637
		4	11672.2	**11327**	12585
		5	12852	**12325**	13400
		6	8597.8	**8272**	8959
		7	15080.2	**14899**	16042
		8	13050.2	**12886**	13768
		9	10273.8	**9920**	11102
	60	0	15487.6	**14761**	15667
		1	15821	**14816**	17200
		2	16356.2	**15854**	16788
		3	20086.6	**19001**	21132
		4	25422.6	**24355**	29102
		5	16792.2	**16253**	18154
		6	19224	**19073**	20107
		7	14256.6	**13593**	15028
		8	16008.2	**15460**	16666
		9	15073.6	**14333**	16001
	70	0	35927.8	**34307**	39086
		1	44626.8	**44117**	46518
		2	30355	**29452**	32751
		3	28845.8	**28351**	30981
		4	34176.8	**33444**	35622
		5	30912.6	**30641**	33754
		6	35234.8	**34693**	38160
		7	31698	**29996**	34744
		8	31782.6	**30693**	36313
		9	29209	**28445**	32874
	80	0	44830.6	**44508**	47060
		1	57408	**56965**	60678
		2	57350.2	**56503**	63546
		3	50011.6	**49246**	54342
		4	66274.6	**65487**	73438
		5	44335	**42449**	49385
		6	51707.2	**50573**	60554
		7	56124.4	**54281**	62469
		8	58725.8	**57664**	63639
		9	46936.2	**46634**	52713
	90	0	76973.6	**75280**	81636
		1	57370.2	**54605**	60940
		2	71412.4	**70111**	78249
		3	56827.6	**55795**	59997
		4	62565.2	**60804**	68553
		5	64097.2	**62985**	68285
		6	61515.6	**60950**	66476
		7	56478.4	**55187**	60331
		8	89471	**88113**	96173
		9	70545.8	**69028**	76427
	100	0	105223.2	**103354**	113156
		1	78856.8	**76374**	86818
		2	94309.2	**91145**	100521
		3	77440	**76859**	83304
		4	69646	**68275**	76942
		5	83592	**81953**	91458
		6	75951.6	**73512**	83883
		7	90988	**88163**	96022
		8	93290	**91048**	99770
		9	89401	**86753**	93511