4as Jornadas Politécnicas de Engenharia
17 e 18 de Novembro de 2004
Instituto Superior de Engenharia do Porto

Programa das Jornadas

Mecânica, Automóvel, Organização

e Gestão Industrial, Energia e Ambiente

Local : Instituto Superior de Engenharia do Porto

Resolução optimizada para 1024 x 768

Uma Meta-heurística para o Problema da Programação de Projectos com Recursos Limitados

Jorge José de Magalhães Mendes[a], José Fernando Gonçalves[b]

[a]Instituto Superior de Engenharia - Instituto Politécnico do Porto
Departamento de Engenharia Informática
Rua Dr. António Bernardino de Almeida, 431
4200-072 Porto, Portugal
e-mail: jjm@isep.ipp.pt

[b]Faculdade de Economia da Universidade do Porto
Rua Dr. Roberto Frias
4200-464 Porto, Portugal
e-mail: jfgoncal@fep.up.pt

RESUMO

Neste artigo propõe-se uma abordagem para a resolução do Problema da Programação de Projectos com Recursos Limitados – ***RCPSP.*** A abordagem combina um algoritmo genético com um novo esquema de geração de planos. O algoritmo genético utiliza uma representação cromossómica baseada em chaves aleatórias. A programação das actividades é feita com recurso a um esquema de geração de planos que se baseia em prioridades e em tempos de espera. As prioridades e os tempos de espera são definidos pelo algoritmo genético. O esquema gerador de planos gera um novo tipo de planos designados por planos activos parametrizados.

O algoritmo é testado num conjunto de problemas padrão retirados da literatura da especialidade e é comparado com outras abordagens. Os resultados computacionais validam o bom desempenho do algoritmo em termos de qualidade da solução.

Palavras-chave: Gestão de Projectos, Programação, Sequenciamento, Algoritmos Genéticos, Chaves Aleatórias.

1. Introdução

O problema da programação ou sequenciamento de um projecto com recursos limitados ***RCPSP*** pode ser descrito da seguinte forma: um projecto consiste num conjunto de actividades $J = \{0, 1, 2, ..., n, n+1\}$ a sequenciar. As actividades *0 e n+1* são fictícias, têm duração nula e correspondem ao início e fim do projecto. As actividades estão interrelacionadas por dois tipos de restrições:

- Restrições de precedência que implicam que cada actividade *j* só pode ser sequenciada após todas as suas actividades precedentes, *P*j, terem sido concluídas;
- Restrições de capacidade que implicam que uma actividade só pode ser sequenciada se existirem recursos disponíveis.

No problema ***RCPSP*** existem $\boldsymbol{K}$ tipos de recursos, representados pelo conjunto $\boldsymbol{K}$={1,...,k}. Cada actividade tem uma duração definida por $\boldsymbol{d_j}$. Entre os instantes de início e de fim, cada actividade $\boldsymbol{j}$ necessita de $\boldsymbol{r_{j,k}}$ de cada recurso $\boldsymbol{k}$ e não pode ser interrompida. Cada tipo de recurso $\boldsymbol{k}$ dispõe de uma capacidade limitada $\boldsymbol{R_k}$ em qualquer instante. Os parâmetros $\boldsymbol{d_j}$, $\boldsymbol{r_{j,k}}$ e $\boldsymbol{R_k}$ são determinísticos; para as actividades de início e fim do projecto $\boldsymbol{d_0}$= $\boldsymbol{d_{n+1}}$=0 e $\boldsymbol{r_{0,k}}$= $\boldsymbol{r_{n+1,k}}$ =0 para qualquer recurso $\boldsymbol{k}$.

A resolução do problema do ***RCPSP*** consiste em determinar os tempos de início e de fim de cada actividade respeitando as relações de precedência e capacidade disponível de cada recurso, por forma a que a duração total do projecto, *makespan*, seja minimizada.

Para ilustrar o problema ***RCPSP***, descreve-se na Figura 1, um exemplo com $\boldsymbol{K}$=2 recursos e $\boldsymbol{n}$=6 actividades. Os limites de capacidade dos recursos 1 e 2 são respectivamente 4 e 2. Para este problema foi obtida uma duração total de 15 unidades de tempo. A Figura 2 ilustra a utilização dos dois recursos para o plano final obtido.

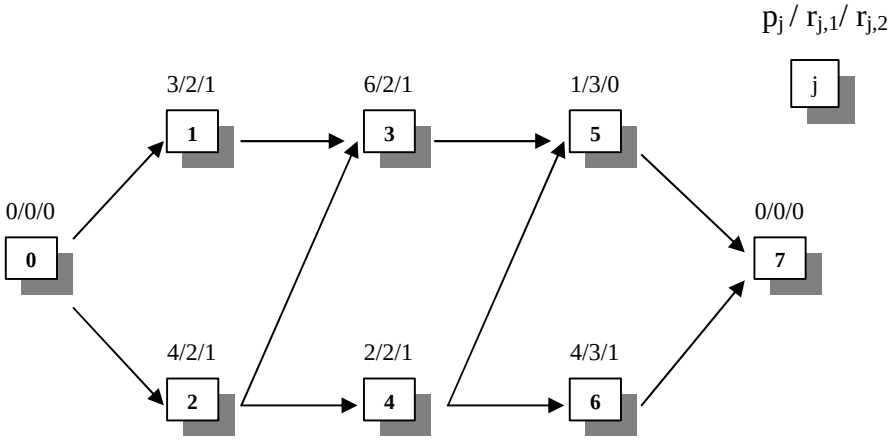


Figura 1 – Exemplo de rede de projecto.

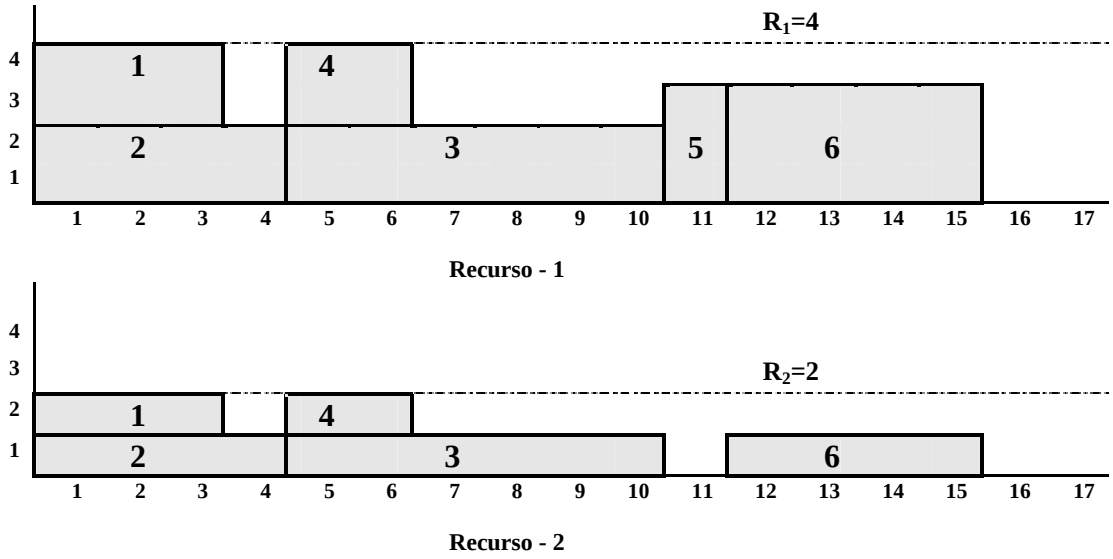


Figura 2 – Exemplo de utilização dos recursos.

Considere-se que F_j representa o tempo de fim da actividade ***j***. Um plano pode ser representado por um vector de tempos de fim das actividades (F_1, F_2, ... , F_n). O conjunto de actividades que estão activas (em processamento) em determinado instante *t* designa-se ***A***(*t*).

O modelo conceptual do problema ***RCPSP*** descrito por Christofides et al. (1987) é o seguinte:

$$\text{Min } F_{n+1} \qquad (1)$$

Sujeito a:

$$F_l \le F_j - d_j \qquad j = 1,\ldots,n+1 \;;\; l \in P_j \qquad (2)$$

$$\sum_{j \in A(t)} r_{j,k} \le R_k \qquad k \in K \;;\; t \ge 0 \qquad (3)$$

$$F_j \ge 0 \qquad j = 1,\ldots,n+1 \qquad (4)$$

A função objectivo (1) minimiza o tempo de fim da última actividade do projecto, designada pela actividade *n+1*, minimizando assim a duração total do projecto. As restrições (2) impõem as relações de precedência entre actividades e as restrições (3) impõem para cada recurso *k* que a utilização resultante das actividades activas em cada instante *t* não exceda a capacidade disponível. Finalmente (4) impõe que as variáveis de decisão sejam não negativas.

2. Revisão da Literatura

Na área dos algoritmos óptimos (em que se obtêm soluções óptimas), um vasto conjunto de abordagens utilizando a técnica do "Branch and Bound" tem sido proposto por vários autores, salientam-se as contribuições de Demeulemeester e Herroelen [10], Sprecher [34], Brucker et al. [8], Klein e Scholl [19] [20], Mingozzi et al. [3] e Mohring et al. [31].

Blazewicz et al. [4] demonstram que o problema ***RCPSP***, é uma generalização do problema clássico do *Job Shop* ***JSP*** o qual pertence à classe de problemas tipo NP-hard. Como consequência, os procedimentos baseados em heurísticas são indispensáveis para a resolução de problemas reais em tempo útil.

As heurísticas que têm vindo a ser utilizadas para a resolução do problema do ***RCPSP*** pertencem a uma das duas classes: a classe dos métodos baseados em regras de prioridade ou a classe das abordagens baseadas em meta heurísticas, Kolisch e Hartmann [24].

Os métodos da primeira classe começam sem solução inicial, isto é, um plano vai sendo construído seleccionando sucessivamente actividades de um conjunto de actividades disponíveis até todas as actividades estarem sequenciadas. Estes métodos são controlados tanto pelos esquemas de sequenciamento como pelas regras de prioridade. As regras de prioridade são utilizadas para estabelecerem a ordem pela qual as actividades são seleccionadas e sequenciadas. Nesta classe salientam-se as contribuições de Alvarez-Valdes e Tamarit [1], Boctor [5], Kolisch e Drexl [23], Kolisch [21] e Kolisch e Hartmann [24].

Os métodos baseados em meta-heurísticas começam com uma solução inicial e tentam melhora-la. A melhoria de uma solução é obtida transformando uma ou várias soluções noutras. Os métodos que têm vindo a ser utilizados são os algoritmos genéticos, *simulated annealing* e o *pesquisa tabu*. Nesta classe salientam-se as contribuições de Leon e Ramamoorthy [27], Baar et al. [2], Hartmann [13] [14], Kolisch e Hartmann [24], Nonobe e Ibaraki [32], Bouleimen e Lecocq [6], Mendes e Gonçalves [29] e Mendes [28].

De referir ainda as revisões de literatura nos trabalhos de Herroelen et al. [17] e Kolisch e Padman [25].

3. Planos Activos Parametrizados

Os planos podem ser classificados em três tipos: os semi-activos, os activos e os não-atrasados. Neste artigo vamos utilizar planos activos parametrizados. Este tipo de planos consiste nos planos nos quais nenhum recurso deixa de ser utilizado para além de um valor pré-definido (designado por tempo de espera), se tiver alguma actividade que pode ter início. O plano óptimo pertence ao conjunto dos planos activos. Contudo, o conjunto dos planos activos é muito extenso e contém planos com grandes tempos de espera e fraca qualidade em termos da solução. Por forma a diminuir o espaço de procura, utiliza-se o conceito de planos activos parametrizados. Através do controle do valor do tempo de espera permitido, podemos aumentar ou diminuir o espaço de soluções ao espaço dos planos não–atrasados.

A secção 4.2 descreve o pseudo-código para a geração de planos activos parametrizados.

4. Nova Abordagem

A nova abordagem apresentada neste artigo combina um algoritmo genético com um procedimento que gera planos activos parametrizados. Em termos gerais a abordagem consiste nas seguintes duas fases:

- **Construção de planos**. Esta fase recorre a um esquema de geração de planos do tipo paralelo modificado para construir planos activos parametrizados. A escolha da actividade a seleccionar em cada iteração do algoritmo é controlada através da prioridade e do tempo de espera associado a cada actividade;
- **Evolução das prioridades e dos tempos de espera**. Esta fase utiliza um algoritmo genético para evoluir (melhorar) as prioridades e tempos de espera utilizadas na fase anterior.

Nas secções seguintes serão apresentados os detalhes da abordagem.

4.1 Algoritmo Genético

Os algoritmos genéticos são baseados nos mecanismos de selecção natural e da genética e foram introduzidos por Holland [16]. Os algoritmos genéticos são normalmente usados para resolver problemas de pesquisa e optimização.

Na teoria da evolução natural sobrevivem os indivíduos mais aptos. A cada indivíduo (cromossoma) de uma população está associado um conjunto de genes, que representa o seu valor ou mérito. Quanto mais apto estiver um indivíduo para a sobrevivência maior será o seu mérito.

Os mecanismos naturais de preservação e evolução de uma população são a sua reprodução e mutação. A reprodução tem por base o cruzamento ("crossover") genético. Através de mutação dos indivíduos é possível incluir diversidade numa espécie.

Os algoritmos genéticos utilizam os seguintes mecanismos:

- **Selecção:** consiste em seleccionar alguns dos melhores cromossomas para a população seguinte;
- **Cruzamento:** consiste no cruzamento de cromossomas preferencialmente de elevado mérito;
- **Mutação:** consiste em incluir diversidade numa população através da alteração dos genes de alguns cromossomas.

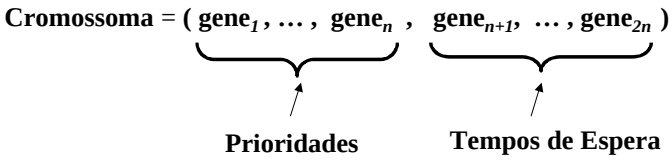
Seguidamente apresenta-se o pseudo-código utilizado no algoritmo genético:

Algoritmo Genético
{
  Gerar população inicial $\boldsymbol{S}_t$
  Avaliar população $\boldsymbol{S}_t$
  Enquanto critério de paragem não satisfeito Repetir
  {
    Seleccionar elementos de $\boldsymbol{S}_t$ a colocar em $\boldsymbol{S}_{t+1}$
    Cruzar elementos de $\boldsymbol{S}_t$ e colocar em $\boldsymbol{S}_{t+1}$
    Mutação de novos elementos e colocar em $\boldsymbol{S}_{t+1}$
    Avaliar nova população $\boldsymbol{S}_{t+1}$
    $\mathbf{S}_t = \boldsymbol{S}_{t+1}$
  }
}

4.1.1 Representação Cromossómica

O algoritmo genético utilizado usa chaves aleatórias como método de representação (codificação) de soluções em vez da codificação binária tradicional. Esta metodologia foi proposta por [2] e consiste na atribuição de números aleatórios, compreendidos entre 0 e 1, aos genes que constituem um cromossoma.

Na abordagem apresentada neste artigo cada cromossoma (solução) é composto por 2*n* genes onde *n* representa o número de actividades. Os primeiros *n* genes são usados para obter as prioridades associadas a cada actividade e os últimos *n* genes são usados para obter os tempos de espera associados a cada iteração, conforme ilustra a figura seguinte.



4.1.2 Descodificação das Prioridades das Actividades

Os primeiros *n* genes são usados para obter as prioridades de cada actividade de acordo a seguinte expressão:

$$\boldsymbol{Prioridade_j = Gene_j.}$$

4.1.3 Descodificação dos Tempos de Espera

Conforme mencionado anteriormente os genes de *n+1* até 2*n* são usados para obter os tempos de espera associados a cada iteração. O tempo de espera, $\boldsymbol{tEspera_g}$, usado para realizar o sequenciamento na iteração $\boldsymbol{g}$, é determinado pela seguinte expressão:

$$\boldsymbol{tEspera_g = Gene_{n+g} \ x \ 1.5 \ x \ MaxTempoProc} \qquad \boldsymbol{g} = 1,\ldots,\boldsymbol{n}$$

na qual ***MaxTempoProc*** é o maior tempo de processamento de entre todas as actividades do projecto.

4.1.4 Estratégia Evolucionária

Inicialmente é gerada aleatoriamente uma população de cromossomas. Esta população será posteriormente transformada através da aplicação dos operadores genéticos.

A reprodução é inicialmente conseguida usando uma estratégia elitista Goldberg [12], na qual os melhores indivíduos de uma população são copiados para a geração seguinte. A vantagem desta estratégia é garantir que a melhor solução vá melhorando sucessivamente de geração para geração.

O operador de cruzamento adoptado é do tipo uniforme parametrizado Spears e DeJong [34], em vez dos tradicionais cruzamentos simples, duplo ou multi-ponto.

O operador de mutação em vez de utilizar a mutação gene a gene com baixa probabilidade, emprega o conceito de imigração. Em cada geração são gerados novos membros a partir da mesma distribuição utilizada para a geração da população inicial. Este processo visa evitar a convergência prematura de uma população.

A Figura 3 exemplifica o processo de transição de uma geração para a seguinte:

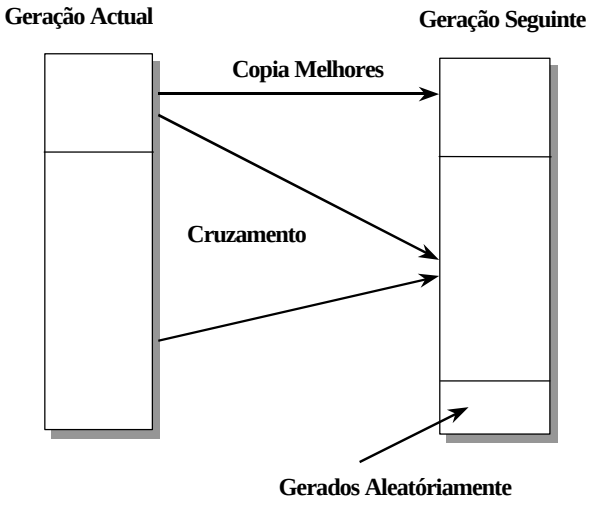


Figura 3 – Transição de geração.

4.2 Esquema de Geração de Planos

O procedimento desenvolvido para construir planos activos parametrizados é baseado num novo esquema de geração de planos que utiliza incrementos de tempo. Para cada iteração ***g,*** existe um tempo t_g que lhe está associado. O conjunto activo de actividades compreende o conjunto de todas as actividades que estão activas em t_g, i.e. $A_g = \{ j \in J \mid F_j - d_j \leq t_g < F_j \}$. A capacidade disponível do recurso k no instante t_g é dada por $RD_k(t_g) = R_k(t_g) - \sum_{j \in A_g} r_{j,k}$. S_g é o conjunto de todas as actividades sequenciadas até à iteração g, e T_g representa os tempos de conclusão das actividades em S_g. Seja $tEspera_g$ o tempo de espera associado à iteração g e E_g o conjunto de todas as actividades que podem ser sequenciadas (cujas predecessoras já foram sequenciadas) no intervalo $[t_g, t_g + tEspera_g]$, i.e.

$$E_g = \{ j \in J \setminus S_{g-1} \mid F_i \leq t_g + tEspera_g \quad (i \in P_j) \}.$$

A Figura 4 descreve o pseudo-código do algoritmo utilizado para gerar planos activos parametrizados.

Inicialização: $g = 1,\ t_1 = 0,\ A_0 = \{0\},\ E_0 = \{0\},\ S_0 = \{0\},\ RD_k(0) = R_k\ \ (k \in K)$

Enquanto $|S_g| < n + 2$ **Repetir**

{

Actualizar E_g

Enquanto $E_g \neq \{\}$ **Repetir**

{

Seleccionar actividade com maior prioridade

$$j^* = \underset{j \in E_g}{\operatorname{argmax}} \left\{ PRIORIDADE_j \right\}$$

Calcular tempo de fim mais cedo (tendo em conta apenas as restrições de precedência)

$$FMC_{j^*} = \max_{i \in P_{j^*}} \{F_i\} + d_{j^*}$$

Calcular tempo de fim mais cedo (tendo em conta as restrições de precedência e de capacidade)

$$F_{j^*} = \min \left\{ t \in \left[FMC_{j^*} - d_{j^*},\ \infty \right] \cap T_g \mid r_{j^*,k} \leq RD_k(t),\ k \in K \mid r_{j^*,k} > 0,\ t \in \left[t,\ t + d_{j^*} \right] \right\} + d_{j^*}$$

Actualizar $S_g = S_{g-1} \cup \left\{ j^* \right\}$, $T_g = T_{g-1} \cup \left\{ F_{j^*} \right\}$

Incrementar a iteração: $g = g+1$

Actualizar $A_g,\ E_g,\ RD_k(t) \mid t \in \left[F_{j^*} - d_{j^*},\ F_{j^*} \right],\ k \in K \mid r_{j^*,k} > 0$

}

Determinar o tempo associado à iteração g

$$t_g = \min \left\{ t \in T_{g-1} \mid t > t_{g-1} \right\}$$

}

Figura 4 – Pseudo-código para gerar planos activos parametrizados.

5. Testes Experimentais

Esta secção apresenta os resultados dos testes experimentais obtidos pelo algoritmo ***GAPS*** (***G**enetic **A**lgorithm for **P**roject **S**cheduling*).

Os testes experimentais foram realizados sobre um conjuntos de problemas designados por J30, J60 e J120. Os conjuntos J30 e J60 consistem cada em 480 problemas e o conjunto J120 consiste em 600 problemas. Cada problema é constituído por um projecto que tem 30, 60 e 120 actividades respectivamente para os conjuntos de problemas J30, J60 e J120. Todos os problemas requerem 4 tipos de recursos. Detalhes dos problemas são descritos em Kolisch et al. [26].

Para a realização dos testes experimentais o algoritmo genético usou a seguinte configuração para todos os problemas:

Tamanho da população:	2 × Número de actividades do problema
Cruzamento:	Probabilidade de cruzamento = 0.7
Selecção:	10% dos melhores cromossomas são copiados para a próxima geração
Mutação:	20% dos cromossomas da população
Mérito:	Makespan (a minimizar)
Sementes:	20
Critério de paragem:	1000 Gerações.

O algoritmo proposto é comparado com os seguintes algoritmos:

Priority Rule Based Sampling Methods

- Schirmer (1998) [33]
- Kolisch, Drexl (1996) [23]
- Kolisch (1996) – Single pass/sampling – WCS [21]
- Kolisch (1996) – Single pass/sampling – LFT [21]
- Kolisch (1995) [22]

Algoritmos Genéticos

- Hartmann (2002) – GA self adapting [14]
- Hartmann (1998) – GA activity list [13]
- Hartmann (1998) – GA random key [13]
- Hartmann (1998) – GA priority rule [13]
- Leon, Ramamoorthy (1995)

Simulate Annealing

- Bouleimen, Lecocq (2003) [6]

Tabu Search

- Nonobi e Ibaraki (2002) [32]
- Baar (1998) [2].

Nas Tabelas 1, 2 e 3 apresentam-se os resultados comparativos do algoritmo ***GAPS*** e dos algoritmos propostos para comparação. Para os problemas do conjunto J30 a medida utilizada foi o desvio médio percentual do *makespan* obtido em relação ao respectivo valor óptimo (D_{OPT}). Para os problemas dos conjuntos J60 e J120 foi utilizado o desvio médio percentual em relação a um limite inferior (D_{LI}) calculado em Hartmann e Kolisch [15].

Na Tabela 1, para os problemas do conjunto J30 o algoritmo ***GAPS*** obteve D_{OPT} = 0.05, valor que iguala o melhor resultado obtido por Nonobe e Ibaraki [32].

Tabela 1 - Resultados experimentais dos desvios médios percentuais para J30.

Algoritmo	SGS	Autores	D_{OPT}
GAPS	***Paralelo Mod.***	***Mendes, Goncalves***	***0,05***
TS – activity list	Heurística especial	Nonobe, Ibaraki (2002) [32]	0.05
GA – self adapting	Série	Hartmann (2002) [14]	0.22
S A-activity list	Série	Bouleimen, Lecocq (2003) [6]	0,23
GA - activity list	Série	Hartmann (1998) [13]	0,25
Sampling - adaptative	Série/Paralelo	Schirmer (1998) [33]	0,44
TS - schedule scheme	Heurística especial	Baar (1998) [2]	0,44
sampling – adaptive	Série/Paralelo	Kolisch, Drexl (1996) [23]	0,52
Single pass/sampling -LFT	Série	Kolisch (1996) [21]	0,53
GA – random key	Série	Hartmann (1998) [13]	0,56
Sampling - random	Série	Kolisch (1995) [22]	1,00
GA – priority rule	Série	Hartmann (1998) [13]	1,12
Single pass/sampling – WCS	Paralelo	Kolisch (1996) [21]	1,28
Single pass/ sampling – LFT	Paralelo	Kolisch (1996) [21]	1,29
Sampling - random	Paralelo	Kolisch (1995) [22]	1,48
GA – problem space	Paralelo	Leon, Ramamoorthy (1995) [26]	1,59

Na Tabela 2, para os problemas do conjunto J60, o algoritmo ***GAPS*** obteve D_{LI} = 11.05. De notar que este valor é melhor em 5% em relação ao valor $\mathbf{D_{LI}}$ = 11.58, obtido pelo algoritmo que mais se aproxima Nonobe e Ibaraki [32].

Tabela 2 - Resultados experimentais dos desvios médios percentuais para J60.

Algoritmo	SGS	Autores	D_{LI}
GAPS	***Paralelo***	***Mendes, Goncalves***	***11.05***
TS – activity list	Heurística especial	Nonobe, Ibaraki (2002) [32]	11.58
GA – self adapting	Série	Hartmann (2002) [14]	11.70
GA - activity list	Série	Hartmann (1998) [13]	11,89
S A-activity list	Série	Bouleimen, Lecocq (2003) [6]	11,90
Sampling - adaptative	Série/Paralelo	Schirmer (1998) [33]	12,59
GA – priority rule	Série	Hartmann (1998) [13]	12,74
sampling – adaptive	Série/Paralelo	Kolisch, Drexl (1996) [23]	13,06
Single pass/sampling – WCS	Paralelo	Kolisch (1996) [21]	13,21
Single pass/ sampling – LFT	Paralelo	Kolisch (1996) [21]	13,23
GA – random key	Série	Hartmann (1998) [13]	13,32
TS - schedule scheme	Heurística especial	Baar (1998) [2]	13,48
GA – problem space	Paralelo	Leon, Ramamoorthy (1995) [27]	13,49
Single pass/sampling -LFT	Série	Kolisch (1996) [21]	13,53
Sampling - random	Paralelo	Kolisch (1995) [21]	14,30
Sampling - random	Série	Kolisch (1995) [22]	15,17

Na Tabela 3, para os problemas do conjunto J120, o algoritmo ***GAPS*** obteve $\boldsymbol{D_{LI}}$ = 33.73. De notar que este valor é melhor em 5% em relação ao valor $\mathbf{D_{LI}}$ = 35.39, obtido pelo algoritmo que mais se aproxima Hartmann [14].

Tabela 3 - Resultados experimentais dos desvios médios percentuais J120.

Algoritmo	SGS	Autores	D_{LI}
GAPS	***Paralelo***	***Mendes, Goncalves***	***33.73***
GA – self adapting	Série	Hartmann (2002) [14]	35.39
TS – activity list	Heurística especial	Nonobe, Ibaraki (2002) [32]	35.85
GA - activity list	Série	Hartmann (1998) [13]	36.74
S A-activity list	Série	Bouleimen, Lecocq (2003) [6]	37.68
Sampling - adaptative	Série/Paralelo	Schirmer (1998) [33]	38.70
Single pass/sampling -LFT	Série	Kolisch (1996) [21]	38.75
Single pass/sampling – WCS	Paralelo	Kolisch (1996) [21]	38.77
sampling – adaptive	Série/Paralelo	Kolisch, Drexl (1996) [23]	40.45
GA – problem space	Paralelo	Leon, Ramamoorthy (1995) [27]	40.69
Single pass/ sampling – LFT	Paralelo	Kolisch (1996) [21]	41.84
GA – priority rule	Série	Hartmann (1998) [13]	42.25
Sampling - random	Paralelo	Kolisch (1995) [22]	43.05
Sampling - random	Série	Kolisch (1995) [22]	47.61

6. Conclusões

Neste artigo apresenta-se uma nova abordagem para o problema da geração de planos de sequenciamento de um projecto com recursos limitados. Esta abordagem combina um algoritmo genético com um esquema de geração de planos.

O algoritmo genético utiliza como alfabeto chaves aleatórias em vez da tradicional codificação binária. O sequenciamento das actividades é realizado com recurso a prioridades e tempos de espera definidos pelo algoritmo genético. Os planos são obtidos através de um esquema de geração de planos activos parametrizados especificamente desenvolvido.

Os testes de desempenho do algoritmo realizados com base nos problemas dos conjuntos J30, J60 e J120 demonstram que o algoritmo ***GAPS*** obtém um excelente desempenho quando comparado com os mais recentes trabalhos científicos na área do sequenciamento de projectos com recursos limitados.

Referências

[1] *Alvarez-Valdés, R. e Tamarit, J.M.. "Heuristic Algorithms for Resource -Constrained Project Scheduling: a Review and an Empirical Analisys". in: R. Slowinski and J. Weglarz Ed., Advances in Project Scheduling, Elsevier, Amsterdam, pp. 113-134 (1989).*

[2] *Baar, T., Brucker, P., Knust, S.. "Tabu-search Algorithms and lower bounds for the resource-constrained project scheduling problem". In: S.Voss, S. Martello, I.Osman, C. Roucairol (Eds.), Meta-heurisitics: Advances and Trends in Local Search Paradigms for Optimization, Kluwer, Boston, pp. 1-8, (1998).*

[3] *Bean, J.C.. "Genetics and Random Keys for Sequencing and Optimization". ORSA Journal on Computing, Vol. 6, pp. 154-160, (1994).*

[4] *Blazewicz, J., Lenstra, J.K., e Rinnooy Kan, A. H. G.. Scheduling subject to resource constraints: Classification and complexity. Discrete Applied Mathematics, 5, pp. 11–24 (1983).*

[5] *Boctor, F. F.. Some Efficient Multi-Heuristics Procedures for Resource-Constrained Project Scheduling. European Journal of Operational Research. Vol 49. pp. 3-13 (1990).*

[6] *Bouleimen, K. e Lecocq, H.. "A new efficient simulated annealing algorithm for the resource-constrained project scheduling problem and its multiple mode version". European Journal of Operational Research. Vol.149(2), pp.268-281, (2003).*

[7] *Brucker, P. e Knust, S.. "A linear programming and constraint propagation-based lower bound for the RCPSP". European Journal of Operational Research 127, pp. 355–362, (2000).*

[8] *Brucker, P., Knust, S. Schoo, A., Thiele, O .. "A branch and bound Algorithms for the resource-constrained project scheduling problem". European Journal of Operational Research 107, pp.272-288, (1998).*

[9] *Christofides, N. Alvarez-Valdés, R. Tamarit, J.M.. "Problem scheduling with resource constraints: A branch and bound approach". European Journal of Operational Research, 29, pp. 262-273, (1987).*

[10] *Demeulemeester, E. e Herroelen, W.. "New benchmark results for the resource-constrained project scheduling problem". Management Science, Vol. 43, pp. 1485-1492, (1997).*

[11] *Demeulemeester, E. e Herroelen, W.. "Project Scheduling – A Research Handbook". Kluwer Academic Publishers, Boston, (2002).*

[12] *Goldberg, D.E.. "Genetic Algorithms in Search Optimization and Machine Learning". Addison-Wesley, (1989).*

[13] *Hartmann, S.. "A competitive genetic Algorithms for resource-constrained project scheduling". Naval Research Logistics 45, pp. 733-750, (1998).*

[14] *Hartmann, S.. "A self-adapting genetic algorithm for project scheduling under resource constraints". Naval Research Logistics 49, pp. 433-448, (2002).*

[15] *Hartmann, S. e Kolisch, R.. "Experimental evaluation of state-of-the-art heuristics for the resource-constrained project scheduling problem". European Journal of Operational Research 127, pp. 394-407, (2000).*

[16] *Holland, J. H.. "Adaptation in Natural and Artificial Systems". University of Michigan Press, Ann Arbor, (1975).*

[17] *Herroelen, W., DE Reyck, B. e Demeulemeester, E.. "Resource-constrained project scheduling: a survey of recent developments". Computers and Operations Research, 25(4), 279-302 (1998).*

[18] *Klein, R.. "Scheduling of Resource-Constrained Projects". Kluwer Academic Publishers, Boston, (2000).*

[19] *Klein, R. e Scholl, A.. "Progress: Optimally solving the generalized resource-constrained project scheduling problem". Working paper, University of Technology, Darmstadt (1998).*

[20] *Klein, R. e Scholl, A. "Scattered branch and bound: An adaptative search strategy applied to resource-constrained project scheduling problem". Working paper, University of Technology, Darmstadt (1998).*

[21] *Kolisch, R.. "Efficient Priority Rules for the Resource-Constrained Project Scheduling Problem". Journal of Operations Management, 14, pp. 179-192 (1996).*

[22] *Kolisch, R.. "Project Scheduling under Resource Constraints". Physica-Verlag Heidelberg, Germany (1995).*

[23] *Kolisch, R. e Drexl, A. "Adaptative search for solving hard project scheduling problems". Naval Research Logistics, Vol. 43, pp. 23-40 (1996).*

[24] *Kolisch, R. e Hartmann, S.. "Heuristic algorithms for solving the resource-constrained project scheduling problem: Classification and computational analysis". In: Weglarz, J. (Ed.), Project Scheduling: Recent Models, Algorithms and Applications. Capítulo 7, pp. 147-178. Dordrecht: Kluwer (1999).*

[25] *Kolisch, R., Padman, R.. "An integrated survey of deterministic project scheduling". The International Journal of Management Science, Vol. 29, pp. 249-272 (2001).*

[26] *Kolisch, R., Schwindt, C., e Sprecher, A.. "Benchmark instances for project scheduling problems". Handbook on recent advances in project scheduling, Capítulo 9, J. Weglarz (ed.), Kluwer, Dordrecht, (1998).*

[27] *Leon, V.J. e B. Ramamoorthy. "Strength and adaptability of problem-space based neighborhoods for resource constrained scheduling". Operations Research Spektrum 17, pp. 173-182 (1995).*

[28] *Mendes, J.M.. "Sistema de Apoio à Decisão para Planeamento de Sistemas de Produção do Tipo Projecto". Tese de Doutoramento, Departamento de Engenharia Mecânica e Gestão Industrial, Faculdade de Engenharia da Universidade do Porto, (2003).*

[29] *Mendes, J.M. e Gonçalves, J.F.. "Um Algoritmo Genético para o Problema do Sequenciamento de Projectos com Recursos Limitados". Revista da Associação Portuguesa de Investigação Operacional, Vol. 23, N. 2, (2003).*

[30] *Mingozzi, A ., Maniezzo, V., Ricciardelli, S. e Bianco, L.. "An exact Algorithm for project scheduling with resource constraints based on a new mathematical formulation". Management Science 44, pp. 714-729, (1998).*

[31] *Mohring, R.H., Schulz, A.S., Stork, F., Uetz, M.. "Solving Project Scheduling Problems by Minimum Cut Computations". Management Science, 49(3), pp.330-350, (2003).*

[32] *Nonobe, K. e T. Ibaraki. "Formulation and tabu search algorithm for the resource constrained project scheduling problem". In C. C. Ribeiro and P. Hansen (Eds.), Essays and Surveys in Meta-heuristics, Capítulo 25, pp. 557–588. Kluwer Academic Publishers, (2002).*

[33] *Schirmer, A.. "Case-Based Reasoning and Improved Adaptative Search for Project Scheduling". Manuskripte aus den Instituten fur Betriebswirtschaftslehre Nº 472, University of Kiel, Germany, (1998).*

[34] *Spears, W.M., e Dejong, K.A.. "On the Virtues of Parameterized Uniform Crossover". Proceedings of the Fourth International Conference on Genetic Algorithms, pp. 230-236, (1991).*

[35] *Sprecher, A.. "Solving the RCPSP efficiently at modest memory requirements". Working paper, University of Kiel, (1997).*