X Congreso Galego de Estatística e Investigación de Operacións
Pontevedra , 3–4–5 de novembro de 2011

SEQUENCING ACTIVITIES IN A PROJECT NETWORK WITH RESOURCE COMPLEMENTARITY – FURTHER RESULTS

Helder C. Silva[1], Anabela P. Tereso[2] and José A. Oliveira[3]

1 IFAM – Instituto Federal de Educação Tecnológica do Amazonas, Manaus, Brazil
2 CITEPE - University of Minho, Guimarães, Portugal
3 Centro Algoritmi - University of Minho, Guimarães, Portugal

ABSTRACT

We address the issue of optimal resource allocation, and more specifically, the analysis of complementarity of resources (primary resource or P-resource and supportive resource or S-resource) to activities in a project. In this paper we present new computational results of a Genetic Algorithm, based in a random keys alphabet.

Keywords : Project Management, Scheduling, Complementarity of resources, Genetic Algorithm

1. INTRODUÇÃO

A Gestão de Projectos é actualmente uma actividade importante em vários sectores da sociedade pós-industrial que inclui agora também a produção de informação. Torna-se fulcral ter-se uma clara percepção das diferentes fases do ciclo de vida de um projecto, dos processos, técnicas e ferramentas, adequados à gestão de um projecto, tendo em atenção a especificidade dos ambientes onde o projecto decorre e a dimensão e complexidade do mesmo.

Para o gestor de projectos é essencial a compreensão dos objectivos de cada projecto em relação a resultados expectáveis, âmbito, custos, tempo e níveis de qualidade e as interligações e dependências entre eles. Não menos importante é também o entendimento da importância da vertente humana (liderança, equipas, conflitos, negociações…) no sucesso de um projecto.

Na gestão de recursos, a selecção e avaliação de pessoal terão em conta o orçamento estabelecido, a disponibilidade dos recursos e ainda o treino e competências específicas. Esta actividade é particularmente crítica, pois pode não ser possível seleccionar as pessoas ideais para trabalhar no projecto, porque o orçamento estabelecido pode não permitir a utilização de profissionais altamente especializados; concomitantemente os profissionais com a experiência apropriada podem não estar disponíveis. Os gestores têm que decidir dentro destas restrições de recursos, resultando que o planeamento de um projecto de software é uma actividade que consome considerável tempo de gestão.

A partição do projecto em tarefas e a estimação do tempo e recursos necessários para completar cada tarefa; a organização das tarefas num formato concorrente para fazer o melhor uso da força de trabalho; e a minimização das dependências entre as tarefas para evitar atrasos, são as funções do gestor de projectos que dependem fortemente da intuição e experiência. Deste modo, a gestão de projectos representa um desafio constante nas empresas.

Este artigo apresenta uma abordagem para a alocação de recursos, em redes de actividades, em condições de complementaridade de recursos, que ocorrem frequentemente no mundo empresarial em projectos de engenharia. A estrutura do algoritmo genético foi discutida em Silva, Oliveira and Tereso (2010) e foram apresentadas as opções tomadas relativamente à configuração do descodificador dos cromossomas e dos operadores genéticos escolhidos.

O conceito de complementaridade, que tem sido discutido sob um ponto de vista económico (Kemer, 1993) pode ser incorporado no domínio da engenharia como um aumento da eficácia de um recurso principal (recurso-*P*) através da adição de um recurso de suporte (recurso-*S*). Aspectos relacionados com a melhoria do desempenho, redução da duração e aumento da qualidade, bem como o efeito do recurso de suporte no custo do projecto, têm sido apresentados por Silva, Tereso e Oliveira (2010).

A questão pode ser formulada como: Que quantidade de recursos principais (recursos-*P*) e de suporte (recursos-*S*) devem ser alocados às actividades do projecto, para atingirmos os melhores resultados, de forma mais económica?

2. DESCRIÇÃO DO PROBLEMA

Considere uma rede de actividades nos vértices (nodos) (AoN) representada por $G = (N, A)$ com o conjunto de vértices $|N| = n$ (representando as "actividades") e o conjunto de arcos $|A| = m$ (representando a relação de precedência entre as actividades). Em geral, cada actividade requer o uso simultâneo de vários recursos (Tereso, Araújo and Elmaghraby, 2008), (Tereso, Araujo, Moutinho and Elmaghraby, 2009a), (Tereso, Araújo, Moutinho and Elmaghraby, 2009b).

Existe um conjunto de recursos "principais", denotados por *P*, com $|P| = r$. Tipicamente, um recurso principal tem várias unidades disponíveis (p.e. trabalhadores, máquinas, processadores, etc.). Adicionalmente, existe um conjunto de recursos de suporte, representados pela letra *S*, com $|S| = s$ (tais como trabalhadores menos qualificados, ou computadores e dispositivos electrónicos, etc.) que devem ser utilizados em conjunto com os recursos principais para aumentar o desempenho destes últimos. O número de recursos de suporte utilizados numa actividade varia em função dos recursos principais necessários para a execução dessa actividade. O impacto sobre o recurso *P* é representado utilizando a variável $0 < v(r_p, s_q) \leq 1$ que indica a fracção pela qual o recurso-*S* (s_q) aumenta a performance do recurso-P (r_p). Tipicamente, $v(r_p, s_q) \in [0{,}1 \; ; \; 0{,}5]$.

Uma actividade normalmente requer uma utilização simultânea de mais de um recurso-*P* para a sua execução. O problema coloca-se então do seguinte modo.

A que nível deve ser utilizado cada recurso principal e que recursos de suporte lhe devem ser adicionados para optimizar um dado objectivo?

O tempo de processamento de uma actividade é dado pelo valor máximo das durações que podem resultar de uma afectação específica a cada recurso (veja a discussão prévia sobre o cálculo da duração considerando múltiplos recursos em Tereso, Araújo and Elmaghraby (2008), Tereso, Araujo, Moutinho and Elmaghraby (2009a) e Tereso, Araújo, Moutinho and Elmaghraby (2009b)).

Neste problema considera-se o custo da utilização dos recursos, uma bonificação por se terminar mais cedo do que o previsto e uma penalização por se terminar mais tarde do que uma data prevista de término para o projecto. Foi já desenvolvido um modelo para minimizar o custo total do projecto, considerando que as actividades podem ser iniciadas logo que sejam sequenciáveis, se houver recursos principais suficientes para as iniciar (Silva, Tereso and Oliveira, 2010). Alguns resultados foram também apresentados utilizando um procedimento baseado na análise da rede de actividades e no conceito "estado" (State Space) (Silva, Tereso and Oliveira, 2010b).

3. ALGORITMO GENÉTICO

Uma vez que o Problema Job Shop (JSP) pode ser visto como um caso particular do problema da programação de projectos com restrições de recursos, do inglês "Resource Constrained Project Scheduling Problem" (RCPSP), iremos estender o Algoritmo Genético (AG) desenvolvido para o JSP (Oliveira, Dias and Pereira, 2010) ao RCPSP, e particularmente para a programação de projectos em redes de actividades sob condições de complementaridade de recursos. O AG é baseado numa representação por chaves aleatórias, a qual permite uma fácil reconfiguração para ser aplicada noutros problemas.

A simplicidade dos AG para modelar problemas complexos e a sua fácil integração com outros métodos de optimização foram os factores considerados para esta escolha. Inicialmente, o algoritmo proposto foi concebido para solucionar a versão clássica do JSP (Oliveira, 2007), mas é possível usar o mesmo método para solucionar outras variantes do JSP (Oliveira, 2006), ou neste caso, para solucionar também uma generalização do JSP que é o caso do RCPSP com complementaridade de recursos.

Uma das características que diferenciam os algoritmos genéticos convencionais é o facto de o algoritmo não tratar directamente com as soluções do problema, mas sim com uma representação da solução, o cromossoma. As manipulações do algoritmo são feitas sobre a representação e não directamente sobre a solução (Goldberg, 1989).

Representamos o problema da programação de projectos num gráfico AoN, porque é semelhante ao gráfico disjuntivo que é usado para representar o JSP (Roy and Sussmann, 1964). Uma actividade só pode ser iniciada se as precedentes estiverem concluídas e se o conjunto de todos os recursos principais necessários estiverem disponíveis. Um projecto tem restrições tecnológicas que determinam uma ordem específica para processar algumas das actividades e é necessário garantir que não haja sobreposição temporal, no processamento de tais

actividades em termos de recursos comuns. Atendendo a esta característica, usamos o conceito de actividade sequenciável (uma actividade que pode ser iniciada) e em cada instante de decisão só é necessário escolher uma actividade do conjunto das actividades sequenciáveis. A escolha da actividade é dirigida pelo algoritmo genético respeitando os alelos existentes nos cromossomas que definem a prioridade de cada actividade.
Neste trabalho é utilizado o alfabeto de chaves aleatórias apresentado por Bean (1994). A característica mais importante das chaves aleatórias é que todos os descendentes gerados pelo cruzamento são soluções admissíveis, quando é usada em conjunto com um procedimento construtivo da solução baseado em todas as operações disponíveis para sequenciamento, e a prioridade é dada pelo alelo da chave aleatória. Através da dinâmica do algoritmo genético, o sistema aprende a relação entre vectores aleatórios e as soluções com bons valores da função objectivo. Outra vantagem da representação de chave aleatória é a possibilidade de se utilizar os operadores genéticos convencionais. Esta característica permite o uso do algoritmo genético noutros problemas de optimização, apenas adaptando algumas rotinas relacionadas com o problema.
O algoritmo genético tem uma estrutura muito simples e pode ser representado no algoritmo da Figura 1. Ela começa com a geração da população e a sua avaliação. Atendendo à aptidão dos cromossomas, os indivíduos são seleccionados para serem progenitores. O cruzamento é aplicado e é gerada uma nova população temporária, que também é avaliada. Comparando-se a aptidão dos novos elementos e dos seus progenitores, a população anterior é actualizada.

```
begin
P ← GerarPopulaçãoInicial()
Avaliar(P)
while condição de paragem não for alcançada do
    P' ← Recombinar(P)   // UX
    Avaliar(P')
    P ← Selecionar(P ∪ P')
end while
```

Figura 1: Algoritmo Genético

4. ALGORITMO CONSTRUTIVO

As soluções são obtidas por um algoritmo, que é baseado no algoritmo de Giffler e Thompson (1960). Enquanto o algoritmo de Giffler e Thompson (1960) pode gerar todos os planos activos para o JSP, o algoritmo construtivo apenas gera o plano de acordo com o cromossoma. Como vantagens desta estratégia, salienta-se a menor dimensão do espaço de soluções, que no entanto inclui a solução óptima e o facto de o procedimento não produzir soluções impossíveis ou desinteressantes do ponto de vista da optimização. Por outro lado, uma vez que as dimensões do espaço de representação e do espaço de soluções são muito diferentes, esta opção pode representar um problema, porque muitos cromossomas podem representar a mesma solução.
O algoritmo construtivo tem n estágios (Figura 2) e em cada estágio uma actividade é sequenciada. Detalhes do algoritmo construtivo podem ser consultados em Silva, Oliveira and Tereso (2010).

Passo 1	Seja $t = 1$ com P_1 sendo nulo. S_1 será o conjunto formado por todas as actividades que não têm predecessoras, ou seja pelas actividades ligadas ao vértice início.
Passo 2	Identificar $j^* = \min_{a_k \in S_k} \{j_k\}$ e identificar A^*. Formar S_t^*.
Passo 3	Seleccionar a actividade a_k^* em S_t^* com o maior valor de alelo de prioridade.
Passo 4	Passar ao próximo estágio por: (1) adição de a_k^* a P_t, criando P_{t+1}; (2) remoção de a_k^* de S_t, criando S_{t+1} por adição (se existirem) a S_t das actividades directamente sucessoras de a_k^* e que tenham todos os seus predecessores em P_{t+1}; (3) aumentar t em 1.
Passo 5	Se existir alguma actividade ainda por sequenciar $(t < N)$, voltar ao *Passo 2*. Senão, Parar.

Figura 2: Algoritmo Construtivo

O formato utilizado é similar ao utilizado por Cheng, Gen e Tsujimura (1996), para apresentar o algoritmo de Giffler e Thompson (1960). No Passo 3, em vez de se usar uma prioridade estabelecida por uma regra de despacho, é usada a informação dada pelo cromossoma. Se o maior valor dos alelos for igual para duas ou mais actividades, então é escolhida aleatoriamente uma dessas actividades.

5. RESULTADOS COMPUTACIONAIS

Para se avaliar a eficácia e a eficiência do algoritmo genético foram desenvolvidas instâncias de teste baseadas em redes apresentadas por Tereso (2002). Foram escolhidas 4 instâncias com 3, 5 e 11 actividades. Apesar de se tratarem de instâncias de pequena dimensão, são suficientes para caracterizar o desempenho do algoritmo genético em termos de evolução da melhor solução encontrada ao longo das iterações, a robustez da pesquisa, entre outras características. A Figura 3 apresenta as redes associadas às instâncias escolhidas.

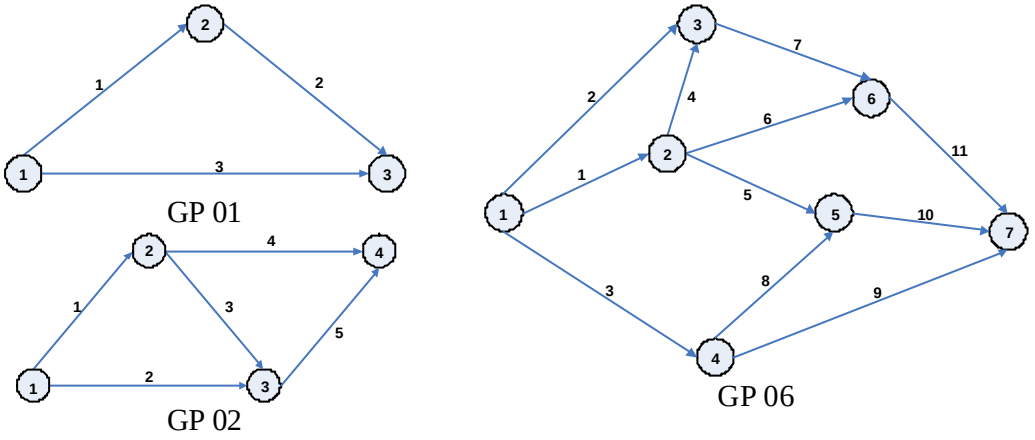


Figura 3: Redes representativas das instâncias

A Tabela 1 resume algumas das experiências computacionais realizadas para estas instâncias. Foram realizadas 5 execuções para cada configuração. Apresentam-se os valores médios e o melhor valor obtido das cinco execuções. Foram testados diferentes dimensões da população, 20, 100 e 300 indivíduos. Pretendeu-se avaliar o desempenho da função de *fitness* estabelecida para o algoritmo genético.

Tabela 1: Resultados computacionais

Instância	População	Iterações	Run1	Run2	Run3	Run4	Run5	Média	Melhor	CPU (s)
GP 01 (3 act)	20	1000	214	214	214	214	214	214,0	214	2
		5000	214	214	214	214	214	214,0	214	
	100	1000	214	214	214	214	214	214,0	214	
		5000	214	214	214	214	214	214,0	214	
	300	1000	214	214	214	214	214	214,0	214	23
GP 02 (5 act)	20	1000	631	631	631	631	641	633,0	631	
		5000	631	631	631	631	631	631,0	631	
	100	1000	631	631	631	631	631	631,0	631	
		5000	631	631	631	631	631	631,0	631	
	300	1000	631	631	631	631	631	631,0	631	35
GP 06_01 (11 act)	20	1000	2584	2548	2529	2427	2444	2506,4	2427	
		5000	2252	2252	2372	2361	2424	2332,2	2252	
	100	1000	2396	2458	2461	2475	2486	2455,2	2396	
		5000	2367	2247	2273	2263	2268	2283,6	2247	
	300	1000	2435	2511	2493	2441	2444	2464,8	2435	93

De um modo geral, verifica-se consistência nos valores obtidos, principalmente nas duas instâncias mais pequenas. O aumento da população permite alcançar melhores valores. Apresentam-se alguns tempos de computação (CPU) em segundos medidos em algumas execuções. São tempos meramente indicativos, pois o código ainda não se encontra optimizado para a rapidez de execução.

De modo a ter-se uma percepção da evolução da solução ao longo das iterações, apresenta-se na Figura 4 o registo da melhor solução e o valor da média de 5 execuções na instância de GP 06_01 ao longo de 25000 iterações, com uma população de 10 indivíduos.

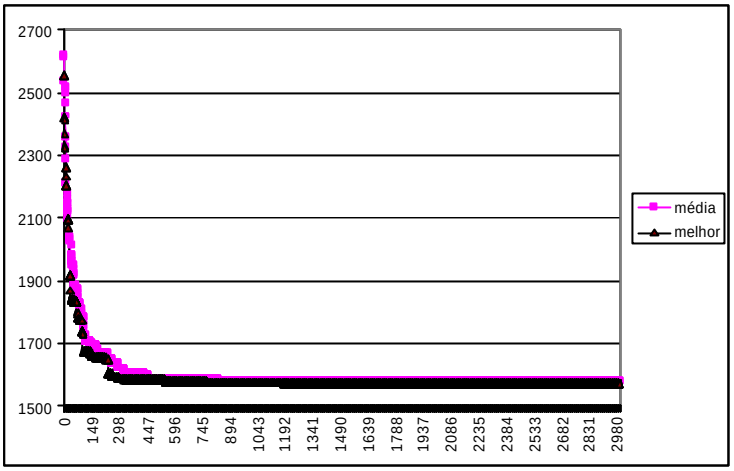
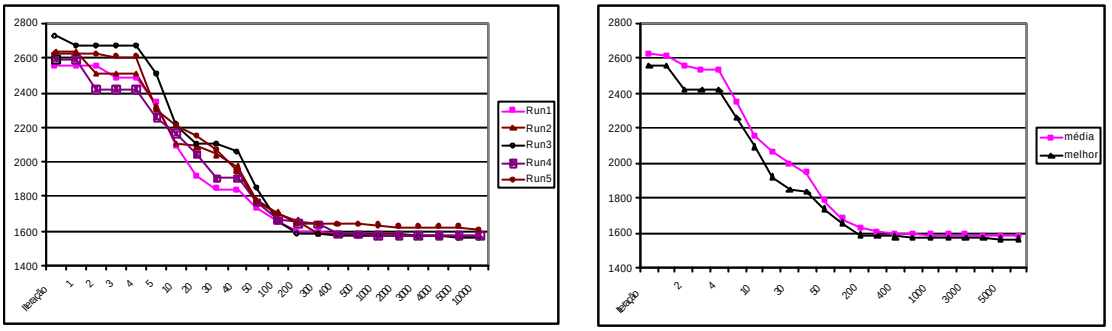


Figura 4: Evolução do desempenho do algoritmo genético

Pode-se verificar que se trata de uma "curva" de aproximação típica neste tipo de heurística. Nos primeiros 20% das iterações obtém-se uma melhoria que corresponde a cerca de 90% da melhoria total que o método faz ao longo das 25000 iterações. A Figura 5 apresenta um resumo dos valores obtidos para as 5 execuções em diferentes iterações entre 1 e 20000.



a) b)

Figura 5: Resumo de 5 execuções

Na Figura 5a) verifica-se um comportamento semelhante e consistente nas 5 execuções. A Figura 5b) apresenta o valor médio e o melhor valor obtido nas 5 execuções. Pode-se verificar a existência de consistência na evolução do valor médio das 5 execuções.

6. CONCLUSÕES

Este artigo apresenta um algoritmo genético desenvolvido para o problema de gestão de projectos com complementaridade de recursos. A importância do problema reside na oportunidade de se desenvolver um sistema que permita não só melhorar a afectação de recursos frequentemente escassos, mas também resultar numa redução das incertezas dentro dos projectos, combinada com o aumento da performance e com a redução do custo do projecto. O método desenvolvido foi testado com algumas redes de actividades e os resultados obtidos permitem demonstrar a sua validade, eficácia e eficiência.

Considerando a viabilidade do modelo proposto, acreditamos que ele pode fornecer ao utilizador uma nova opção de planeamento para determinar a melhor combinação de recursos e o menor custo do projecto, melhorando a capacidade de planeamento das empresas.

Na sequência deste estudo pretendemos testar o procedimento apresentado num conjunto de instâncias de maior dimensão e desenvolver alguns operadores genéticos específicos para o problema, que serão implementados essencialmente sob a forma de mutação, no sentido de aumentar a diversidade da população e evitar a convergência prematura que se verifica ao fim de 20% das iterações.

REFERENCIAS

Bean, J. (1994). Genetics and random keys for sequencing and optimization. ORSA Journal on Computing, 6, pp. 154–160.

Cheng, R., Gen, M. and Tsujimura, Y. (1996). A tutorial survey of job-shop scheduling problems using genetic algorithms - I. Representation, Computers & Industrial Engineering, 30, pp. 983-997.

Giffler, B. and Thompson, G. L. (1960). Algorithms for solving production scheduling problems. Operations Research, 8, pp. 487-503.

Goldberg, D. E. (1989). Genetic Algorithms in Search. Optimization and Machine Learning. Addison-Wesley, Reading, MA.

Kemer, M. (1993). The O-Ring Theory of Economic Development. The Quarterly Journal of Economics, 108, 551-575.

Oliveira, J. (2006). A genetic algorithm with a quasi-local search for the job shop problem with recirculation. Applied Soft Computing Technologies: The Challenge of Complexity, (pp. 221-234). Springer, Berlin / Heidelberg.

Oliveira, J. (2007). Scheduling the truckload operations in automatic warehouses. European Journal of Operational Research, 179, 723-735.

Oliveira, J., Dias, L., and Pereira, G. (2010). Solving the Job Shop Problem with a random keys genetic algorithm with instance parameters. 2nd International Conference on Engineering Optimization. Lisbon – Portugal.

Roy, B. and Sussmann, B. (1964). Les problèmes d'ordonnancement avec contraintes disjonctives. Paris.

Silva, H., Oliveira, J. and Tereso, A. (2010). Um Algoritmo Genético para Programação de Projectos em Redes de Actividades com Complementaridade de Recursos. Revista Ibérica de Sistemas y Tecnologías de la Información. 6, 59-72.

Silva, H., Tereso, A. and Oliveira, J. (2010). On Resource Complementarity. 3rd International Conference on Information System, Logistics and Supply Chain. Casablanca - Morocco.

Silva, H., Tereso, A. and Oliveira, J. (2010b). On Resource Complementarity in Activity Networks - Further Results. 2nd International Conference on Engineering Optimization. Lisbon - Portugal.

Tereso, A.P. (2002). Project Management - Adaptive Resource Allocation in Multimodal Activity Networks. PhD Thesis. University of Minho - Portugal.

Tereso, A., Araújo, M. and Elmaghraby, S. (2008). Project management: multiple resources allocation. International Conference on Engineering Optimization. Rio de Janeiro - Brazil.

Tereso, A. P., Araujo, M., Moutinho, R. and Elmaghraby, S. (2009a). Duration Oriented Resource Allocation Strategy on Multiple Resources Projects under Stochastic Conditions. International Conference on Industrial Engineering and Systems Management. Montreal - Canada.

Tereso, A., Araújo, M., Moutinho, R. and Elmaghraby, S. (2009b). Quantity Oriented Resource Allocation Strategy on Multiple Resources Projects under Stochastic Conditions. International Conference on Industrial Engineering and Systems Management. Montreal - Canada.