Instituto Superior de Engenharia do Porto

Relatório do Sprint C de ALGAV

Turma 3DB _ Grupo 007
1211148 _ Joana Perpétuo
1211016 _ João Rodrigues
1221693 _ Hugo Silva
1201551 _ António Martingo

Índice

Índice de Figuras

Índice de Tabelas

Estudo da complexidade da geração de todas as sequências de tarefas e escolha da melhor

O objetivo é fazer um estudo da viabilidade da utilização de um algoritmo que gera todas as combinações possíveis de ordenação de tarefas, de forma a chegar à melhor combinação possível, tendo em conta os critérios de otimização definidos, e comparar esses resultados com um algoritmo genético, que não necessariamente obtém a melhor solução, mas aproxima-se o mais possível à solução ótima.

Figura 1- Algoritmo de geração de todas as combinações de tarefas e escolha da melhor

```
% Calcula a duração total de uma sequência de tarefas
duracao_total([T], Duracao) :-
    tarefa(T, Tempo, _, _),
    Duracao is Tempo.
duracao_total([T1, T2 | Ts], Duracao) :-
    tarefa(T1, Tempo, _, _),
    transicao(T1, T2, TempoTransicao),
    duracao_total([T2 | Ts], DuracaoResto),
    Duracao is Tempo + TempoTransicao + DuracaoResto.

% Encontra a sequência de tarefas com a menor duração total
melhor_sequencia(Seq, MelhorDuracao,TSol) :-
    get_time(Ti),
    findall(T, tarefa(T, _, _, _), Tarefas),
    permutation(Tarefas, Seq),
    duracao_total(Seq, Duracao),
    \+ (permutation(Tarefas, OutraSeq),
        duracao_total(OutraSeq, OutraDuracao),
        OutraDuracao < Duracao),
    MelhorDuracao = Duracao,
    get_time(Tf),
    TSol is Tf-Ti.
```

Para esse efeito, será executado o algoritmo do algoritmo referido acima, para um conjunto de tarefas, fazendo variar o número de tarefas e registando o tempo de execução do algoritmo para cada caso.

- Serão feitos 5 testes para cada número de tarefas, de forma a ser obtida uma média do tempo de execução;
- De seguida, será feito um gráfico com o tempo de execução médio em função do número de tarefas;
- Para comparação, o mesmo processo será efetuado para o algoritmo genético.

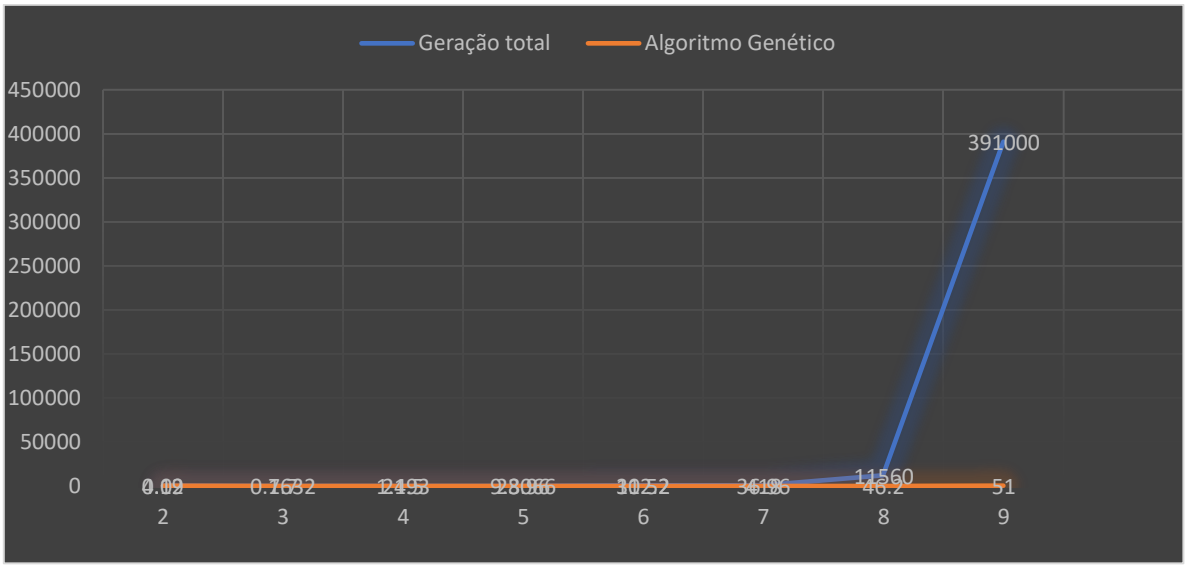
Tabela 1- Relação nrº tarefas-tempo do algoritmo de geração de todas as soluções e escolha da melhor

Nº de tarefas	Execução 1	Execução 2	Execução 3	Execução 4	Execução 5	Média	Custo
2	~0.105ms	~0.095ms	~0.069ms	~0.106ms	~0.075ms	0.09ms	9
3	~0.126ms	~0.074ms	~0.204ms	~0.118ms	~0.294ms	0.1632ms	12
4	~1.111ms	~1.014ms	~1.210ms	~1.060ms	~1.570ms	1.193ms	17
5	~9.211ms	~9.001ms	~11.612ms	~9.645ms	~9.564ms	9.8066ms	21
6	~120ms	~77ms	~119ms	~95ms	~100ms	102.2ms	27
7	~415ms	~429ms	~438ms	~410ms	~398ms	418ms	31
8	~10.6s	~11.0s	~13.4s	~11.3s	~11.5s	11.56s	35
9	~388s	~392s	~393s	~397s	~386s	391s	42

Tabela 2- Relação nrº tarefas - tempo do algoritmo genético (fixado a 10 gerações, 10 indivíduos, 90% cruzamento, 10% mutação)

Nº de tarefas	Execução 1	Execução 2	Execução 3	Execução 4	Execução 5	Média	Menor Custo/ Maior Custo
2	~4.6ms	~3.7ms	~4.2ms	~4.0ms	~4.1ms	4.12ms	9/9
3	~14.7ms	~6.9ms	~5.7ms	~6.2ms	~5.0ms	7.7ms	12/20
4	~23.8ms	~24.8ms	~26.6ms	~24.9ms	~22.4ms	24.5ms	17/60
5	~32.9ms	~35.0ms	~18.6ms	~30.9ms	~29.4ms	23.96ms	21/33
6	~34.5ms	~33.6ms	~27.9ms	~31.3ms	~30.3ms	31.52ms	29/36
7	~21.9ms	~41.3ms	~36.1ms	~42.1ms	~43.4ms	36.96ms	34/47
8	~51.2ms	~52.2ms	~48.0ms	~48.1ms	~31.5ms	46.2ms	39/75
9	~35.4ms	~57.0ms	~57.0ms	~48.7ms	~56.9ms	51ms	48/79

Figura 2- Representação gráfica da relação nrº tarefas – tempo (ms)



Como se pode ver pelo gráfico e pelos valores na tabela, este algoritmo começa por ser bastante rápido para um número pequeno de tarefas, no entanto, à medida que este número aumenta, o tempo aumenta exponencialmente, chegando a um ponto (cerca de 9 tarefas) em que já não é viável utilizar este algoritmo, pois este chega a demorar, no pior caso obtido, cerca de 3.2 minutos, tempo este que não é aceitável a nível aplicacional. Por oposição, o algoritmo genético mantém sempre um nível estável e baixo para os casos estudados.

Para efetuar a escolha do algoritmo a utilizar, é necessário perceber quais os requisitos do problema. Se forem sempre usados números reduzidos de tarefas, é preferível usar o algoritmo de geração de todas as soluções e escolha da melhor, uma vez que é rápido e produz a melhor solução sempre. Para problemas com maior número de tarefas (>9) será quase obrigatório usar o algoritmo genético.

Relativamente à análise de complexidade na notação big-Oh, verificamos no predicado ***melhor_sequencia/3***, 4 momentos de impacto:

- ***findall(T, tarefa(T, _, _, _), Tarefas) :*** De complexidade O(n), em que n é o número de tarefas. Faz a recolha de todas as tarefas da base de conhecimento;
- ***permutation(Tarefas, Seq):*** De complexidade O(n!), em que n é o número de tarefas. Cria todas as sequências possíveis de tarefas por permutações;
- ***duracao_total(Seq, Duracao):*** De complexidade O(n), em que n é o número de tarefas. Obtém a duração total de uma sequência de n tarefas;
- ***\+ (permutation(Tarefas, OutraSeq), duracao_total(OutraSeq, OutraDuracao), OutraDuracao < Duracao):*** De complexidade O((n!)^2), uma vez que verifica as restantes permutações, calcula a duração para cada uma e verifica se é menor.

Com isto, é possível aferir que na notação Big-Oh, o predicado ***melhor_sequencia/3*** tem uma complexidade máxima de O((n!)^2), que é bastante alta, confirmando o que foi provado empiricamente na anterior demonstração do crescimento do tempo de execução relativamente ao aumento do número de tarefas.

Estudo do estado da arte na geração de trajetórias de robots

A evolução tecnológica, no âmbito da indústria de robótica tem tido um grande impacto no desenvolvimento de sistemas móveis autónomos. Nos últimos anos, têm surgido projetos, sendo alguns implantados em negócios, robots com a capacidade de fazer percursos sem apoio do ser humano, por exemplo, robots de entrega de pedidos em restaurantes, de movimentação de produtos em armazém, hospitais e condução automóvel autónoma.

Um dos problemas fundamentais num destes sistemas autónomos é a navegação. Para que esta seja eficaz, é essencial assegurar os seguintes processos (Maia, 2019):

- Perceção do espaço de trabalho;
- Localização e construção de um mapa: onde são mapeados os obstáculos e determinada a posição e orientação do robot;
- Geração da trajetória: onde se encontra a melhor trajetória a ser seguida pelo robot;
- Controlo do movimento.

O estudo em questão reflete sobre o estado da arte da geração de trajetórias. O objetivo para um robot autónomo deve ser deslocar-se da posição atual para outro local e, eventualmente, para a posição final pretendida, ao mesmo tempo evitando quaisquer obstáculos e sendo o mais eficiente possível (Maia, 2019).

Dentro de linhas gerais, os métodos de planeamento destas trajetórias seguem um dos seguintes métodos (Correia, 2013):

- **Controlo ótimo**: consiste na procura de uma lei de controlo para um dado sistema que obedeça a um critério de otimização. Para tal, procede-se à minimização ou maximização de uma função designada por custo;
- **Procura em grafos**: supondo que foi obtido um grafo através de algum dos métodos de discretização do espaço, posteriormente pode ser utilizado um método de pesquisa de caminho, como Pesquisa Primeiro em Largura, o Método de Pesquisa Primeiro em Profundidade, Dijkstra, A* e variantes, D* e variantes. As limitações destes métodos são o facto de não garantirem que o algoritmo seja completo, nem que o caminho seja possível.
- **Campo de potencial**: um Método de Campo de Potencial modeliza o robot como um ponto sujeito a um campo artificial de potencial resultante do somatório das forças existentes no espaço. Estas forças são provenientes dos obstáculos (forças repulsivas) e da posição onde se pretende que este chegue (força atrativa). As limitações deste método são o facto de estar sujeito à existência de mínimos locais e a possibilidade de haver oscilações.

Em 2022, as abordagens e tendências do desenvolvimento de algoritmos de geração de trajetórias para robots incluem (Chat GPT, 2023)*:

1. **Machine Learning (ML) e Deep Learning (DL):** O uso de técnicas de ML e DL têm se destacado na geração de trajetórias para robôs. Redes neurais, em particular, têm sido empregues para aprender padrões complexos de movimento e otimizar trajetórias com base em dados históricos.
2. **Otimização:**
   - → Otimização Convexa: Métodos de otimização convexa são amplamente utilizados para resolver problemas de trajetória. Eles garantem eficiência computacional e são aplicáveis a sistemas dinâmicos complexos;
   - → Algoritmos Genéticos e Algoritmos Evolutivos: Essas abordagens baseadas em evolução são usadas para otimizar trajetórias de maneira iterativa, explorando soluções num espaço de busca.
3. **Controle Preditivo e Modelagem Dinâmica:** Modelos dinâmicos precisos do robot são combinados com estratégias de controle preditivo para gerar trajetórias que levam em consideração as dinâmicas complexas do sistema.
4. **Navegação Autónoma:**
   - → Mapeamento Simultâneo e Localização (SLAM): Em ambientes desconhecidos, sistemas de SLAM são integrados à geração de trajetória para permitir que o robot mapeie o seu ambiente enquanto se desloca;
   - → Tomada de Decisão Autônoma: Algoritmos de tomada de decisão autónoma são implementados para que os robots possam reagir dinamicamente a mudanças no ambiente e ajustar as suas trajetórias em tempo real.
5. **Colaboração e Coordenação de Múltiplos Robôs:**
   - → Algoritmos de Colaboração: À medida que a investigação avança, há um interesse crescente na geração de trajetórias para sistemas multi-robots que cooperam entre si;
   - → Estratégias de Coordenação: A coordenação eficiente entre robots é essencial, e a geração de trajetórias considera cada robot como parte de um sistema maior.
6. **Robótica Baseada em Tarefas:**
   - → Programação por Demonstração: Os robots aprendem a realizar tarefas observando demonstrações humanas, e a geração de trajetórias é adaptada com base nessas demonstrações;
   - → Hierarquia de Tarefas: Trajetórias são geradas em diferentes níveis de abstração para atender tarefas específicas.

*O prompt utilizado para a geração da informação apresentada foi: “estado da arte na geração de trajetórias de robots”.

Referências

Correia, M. M. (13 de Fevereiro de 2013). *Planeamento e controlo otimizado de trajetórias de robots aquáticos.* Obtido de up: https://paginas.fe.up.pt/~ee08218/tese/Docs/pdi.pdf

Maia, T. (Outubro de 2019). *Desenvolvimento de Sistema de Controlo de Movimento para Plataformas Móveis Autónomas.* Obtido de uminho: https://repositorium.sdum.uminho.pt/bitstream/1822/64344/1/57126-Desenvolvimento_de_Sistema_de_Controlo_de_Movimento_para_Plataformas_M%C3%B3veis_Aut%C3%B3nomas.pdf