

Análise de Algoritmos Aproximativos para o Problema da Mochila

Disciplina: Análise e Projeto de
Algoritmos

Antonio Heitor Gomes Azevedo

Deivison Ryan Brito Tavares

Gustavo Yuji Virgolino Nishimura

Heitor Yasuo Yamamoto

1. Introdução

Objetivo: Explorar a aplicação de algoritmos de aproximação para a resolução de problemas intratáveis.

- ✓ **Problema:** Problema da Mochila (Knapsack Problem - KP), variação 0/1.
- ✓ **Meta:** Maximizar o valor total dos itens sem exceder a capacidade de peso.
- ✓ **Motivação:** A solução exata é computacionalmente custosa para grandes instâncias.
- ✓ **Abordagem:** Comparação entre **Particle Swarm Optimization (PSO)** e **Simulated Annealing (SA)**.

2.1. Classes de Problemas

P (Polynomial Time)

Problemas resolvidos por máquina determinística em tempo polinomial. Soluções rápidas e eficientes.

NP (Nondeterministic Polynomial)

Soluções podem ser *verificadas* em tempo polinomial. Note que $P \subseteq NP$.

NP-Completo (NP-C)

Os mais difíceis de NP. Se resolvermos um NP-C em tempo polinomial, resolvemos todos em NP.

NP-Difícil (NP-Hard)

Pelo menos tão difíceis quanto os NP-C. Não precisam estar em NP (ex: problemas de otimização onde verificar o ótimo é difícil).

2.2. O Problema da Mochila na Hierarquia

- ✓ **Versão de Decisão:** "Existe subconjunto com valor $\geq K$ e peso $\leq W$?" \rightarrow **NP-Completo**.
- ✓ **Versão de Otimização:** Maximizar o valor \rightarrow **NP-Difícil**.
- ✓ **Justificativa:** Algoritmos exatos (força bruta) possuem complexidade $O(2^n)$, tornando-se inviáveis com o crescimento de n .
- ✓ Isso torna obrigatório o uso de algoritmos aproximativos para instâncias reais de grande porte.

NP-Completo

Difícil de resolver,
mas fácil de verificar



NP-Difícil

Difícil de resolver
e difícil verificar



Classe	Resolver rápido?	Verificar rápido?
NP-Completo	Não	Sim
NP-Difícil	Não	Não

2.3. Redução Polinomial ($A \leq_p B$)

Técnica para provar a intratabilidade:

- ✓ Transformar uma instância do problema A em uma instância do problema B em tempo polinomial.
- ✓ **Prova:** A NP-Compleitude da Mochila é provada via redução da Cobertura Exata (**Exact Cover**) ou **Partição**.
- ✓ **Implicação:** Resolver a Mochila eficientemente implicaria em resolver todos os problemas da classe NP.

Resumindo: a solução perfeita é cara demais, então usamos métodos rápidos que chegam bem perto do ótimo.

3.1. Metodologia: Configuração (Python 3)

Particle Swarm (PSO)

População	30 Partículas
Iterações	200
Inércia (w)	0.7
Cognitivo (c1)	1.5
Social (c2)	1.5

Simulated Annealing (SA)

Temp. Inicial (T0)	100
Resfriamento (α)	0.99
Iterações	3000

3.2. Análise Assintótica (Big-O)

PSO: $O(I \cdot P \cdot n)$

Variáveis: I (Iterações), P (Partículas), n (Itens).

Justificativa: A complexidade é dominada pela atualização de todas as partículas em cada iteração, percorrendo o vetor de itens.

Simulated Annealing: $O(I \cdot n)$

Variáveis: I (Iterações), n (Itens).

Justificativa: Em cada iteração, gera-se um vizinho e avalia-se a função objetivo para uma solução de tamanho n.

4.1. Tabela Comparativa (100 Execuções)

Instância	Métrica	PSO	SA	Obs.
Pequena (n=10)	Melhor Valor	344	344	Empate
Tempo Médio	0.065s	0.012s	SA 5x mais rápido	
Média (n=30)	Melhor Valor	1001	952	PSO +5.1%
Tempo Médio	0.158s	0.019s	SA 8x mais rápido	
Grande (n=100)	Média Valor	2928.32	2459.50	PSO +19%
Desvio Padrão	67.64	271.41	PSO mais estável	
Tempo Médio	0.525s	0.048s	SA 10x mais rápido	

4.2. Discussão Crítica

- ✓ **Qualidade:** O PSO foi superior em consistência. Na instância grande, a média foi quase 20% maior. O baixo desvio padrão indica robustez.
- ✓ **Tempo:** O SA confirmou sua complexidade $O(l \cdot n)$, sendo até 10x mais rápido. Porém, a velocidade custou a qualidade (alto desvio padrão).
- ✓ **Convergência:** Para problemas NP-Difíceis, o custo de tempo do PSO (0.5s) é desprezível e justifica o ganho de qualidade. O SA é preferível apenas em sistemas de tempo real crítico.

5. Conclusão

- ✓ **Aproximação:** Ferramenta vital. Buscamos solução "boa o suficiente" em tempo polinomial, pois a exata é inviável.
- ✓ **PSO (Equilíbrio):** Apresentou o melhor equilíbrio geral. Oferece alta qualidade com estabilidade.
- ✓ **SA (Velocidade):** Destacou-se pela velocidade extrema, mas com menor confiabilidade nos resultados médios.

6. Referências Bibliográficas

- ✓ CORMEN, T. H. et al. *Introduction to Algorithms*. 3rd Edition. MIT Press, 2009.
- ✓ GAREY, M. R.; JOHNSON, D. S. *Computers and Intractability: A Guide to the Theory of NP-Completeness*. W. H. Freeman, 1979.
- ✓ KENNEDY, J.; EBERHART, R. "Particle Swarm Optimization". *Proceedings of ICNN'95*, 1995.
- ✓ KIRKPATRICK, S. et al. "Optimization by Simulated Annealing". *Science*, 1983.