PSO APLICADO AO PROBLEMA DE TABELA HORÁRIO

Filipe Gomes Arante de Souza

Metaheurísticas 2023/2
Prof. Maria Claudia Silva Boeres

Estrutura da Apresentação

- Introdução;
- Revisão do PSO;
- Apresentação do Problema de Tabela Horário;
- Adaptação do PSO para o PHTU;
- Apresentação dos resultados;

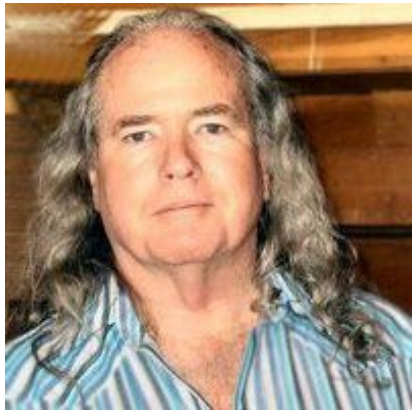
Introdução

- Criado por James Kennedy e Russel Eberhart em 1995;

- Surgiu de estudos oriundos de simulações de modelos sociais simplificados de um bando de pássaros;

- Aplicável em problemas de telecomunicações, controle, mineração de dados, design, otimização combinatória, sistemas de energia, processamento de sinais, ...;

- Um dos algoritmos populacionais mais conhecidos e amplamente utilizados atualmente;



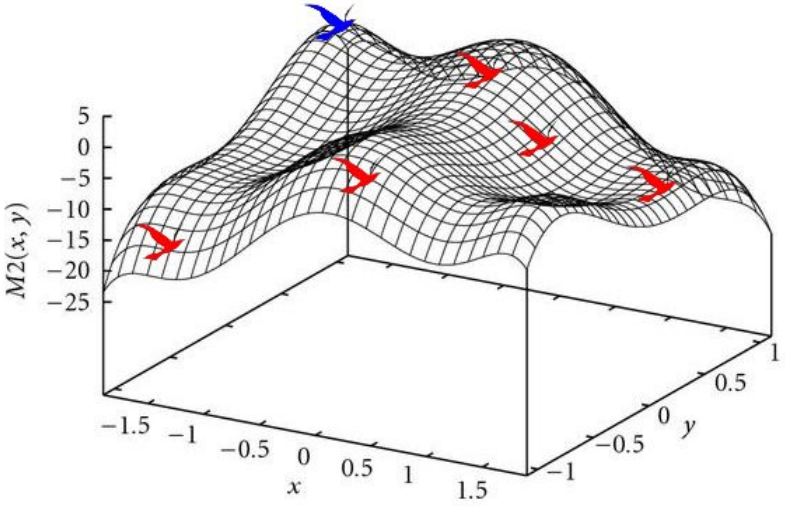
James K, Psicólogo Social

Russel E, Eng. Elétrico

Otimização por Enxame de Partículas

- Procura a solução ótima num determinado espaço de busca através da troca de informações entre indivíduos de uma população;

- As partículas possuem memória da melhor posição que elas já estiveram e conhecem a melhor posição que o bando inteiro já encontrou;



Movimento das Partículas

$$v_{k+1} = w \cdot v_k + c_1 \cdot r_1 \cdot (p_{best_k} - x_k) + c_2 \cdot r_2 \cdot (g_{best} - x_k)$$

$$x_{k+1} = x_k + v_k$$

``A cada iteração do algoritmo as partículas tentam se aproximar um pouco do seu melhor local e do melhor local. Portanto, elas funcionam como se fossem atratores para regiões do espaço potencialmente promissoras.``

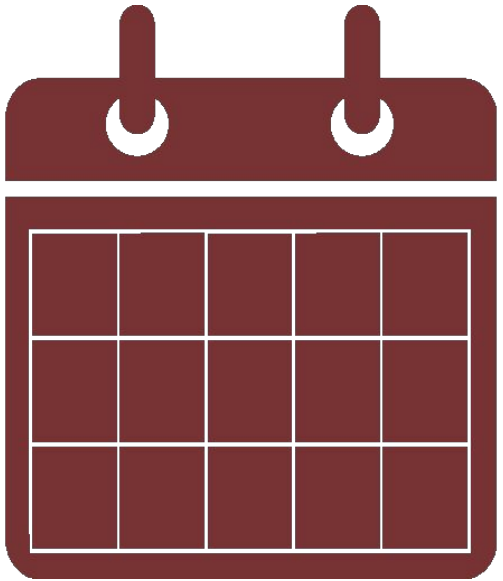
(lembre-se disso)

O Problema de Tabela Horário: Formulação ITC-2007

- Alocar aulas de disciplinas num conjunto finito de horários e salas;

- Cada matéria possui restrições de currículo, professor e indisponibilidades;

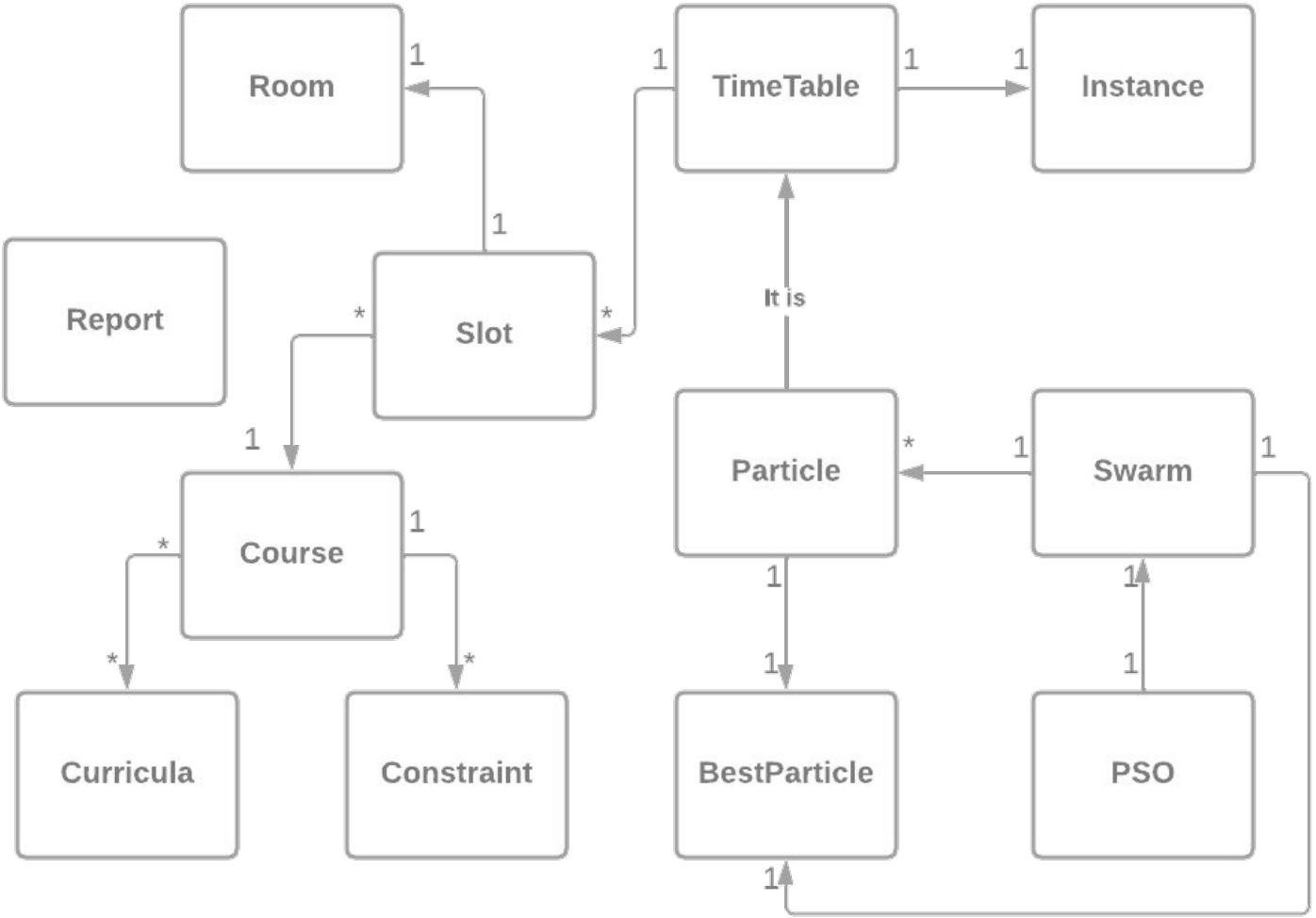
- Objetivo: eliminar as restrições fortes e minimizar as restrições fracas;



Adaptação do PSO para o PHTU

O algoritmo original é para problemas de otimização contínua... e agora?

MODELAGEM


Room
TimeTable
Instance
Report
Slot
It is
Particle
Swarm
Course
Curricula
Constraint
BestParticle
PSO
1
*

Geração de uma Partícula [de Almeida Segatto 2017]

- Abrir imagem do artigo, não coube no slide... :(

Geração da População Inicial

Algorithm 2 Population Constructor

```
1: function POPULATION_CONSTRUCTOR(A, B, α, N)
2:     ▷ A is the set of all classes to alloc
3:     ▷ B is the set of all slots in timetable
4:     ▷ α is a real number between 0 and 1
5:     ▷ N is the size of population
6:
7:     P ← ∅
8:
9:     for ∀i ∈ [1, N] do
10:        p ← GRASP_CONSTRUCTOR(A, B, α)
11:
12:        if p ≠ ∅ then
13:            P ← P ∪ {p}
14:        end if
15:    end for
16:
17:    return P
18: end function
```

Operador: W Rand Mutate [Chu et al. 2006]

Algorithm 3 W Rand Mutate

```
1: function RAND_MUTATE(X, W)
2:     ▷ X is a particle
3:     ▷ W is a positive integer
4:
5:     for ∀i ∈ [1, W] do
6:         for ∀j ∈ [1, 10] do ▷ Try 10 times a successful swap
7:             Select two random slots S1 and S2 from X
8:             Swap the allocated courses in S1 and S2
9:
10:            if X is feasible then
11:                break
12:            else
13:                Undo the swap of allocated courses in S1 and S2
14:            end if
15:        end for
16:    end for
17: end function
```

Operador:
C Rand Change
[Chu et al. 2006]

Algorithm 4 C Rand Change

```
1: function RAND_CHANGE(X, R, C)
2:     ▷ X is a particle to be modified
3:     ▷ R is a reference particle
4:     ▷ C is a positive integer
5:
6:     for ∀i ∈ [1, C] do
7:         for ∀j ∈ [1, 10] do ▷ Try 10 times a successful swap
8:             Select a random slot S_R from R
9:             Get the respective slot S_X of S_R in X
10:            Find some slot S'_X from X that contains the same allocated course of S_R
11:            Swap the allocated courses in S_X and S'_X
12:
13:            if X is feasible then
14:                break
15:            else
16:                Undo the swap of allocated courses in S_x and S'_x
17:            end if
18:        end for
19:    end for
20: end function
```

Nova Equação do PSO

$$S_k^{i+1} = rand\_mutate(X_k^i, W)$$

$$W_k^{i+1} = rand\_change(S_k^{i+1}, P_k^i, C_1)$$

$$X_k^{i+1} = rand\_change(W_k^{i+1}, G^i, C_2)$$

Execução das Instâncias

- **21 instâncias** disponibilizadas;
- Executadas em três computadores:
- Benchmark **204** segundos: **comp01-05** e **comp16-21**;
- Benchmark **228** segundos: **comp01-06**;
- Benchmark **216** segundos: **comp11-15**;
- Instâncias mais difíceis: **comp05** e **comp19**;

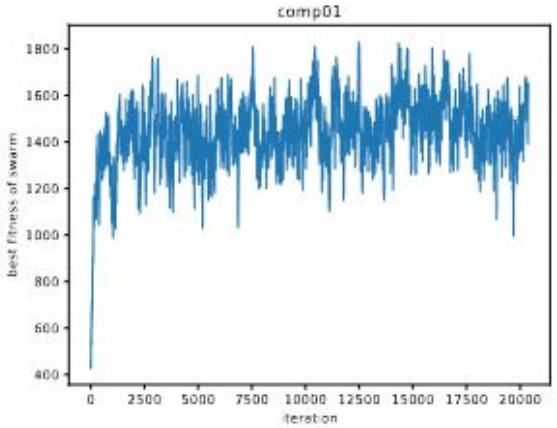
- Parâmetros escolhidos para o PSO:

$$W = 1, C_1 = 1, C_2 = 1$$

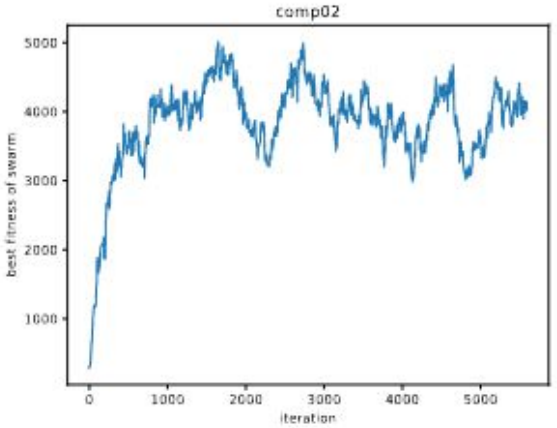
Resultados Computacionais

- Abrir imagem do artigo, não coube no slide... :(

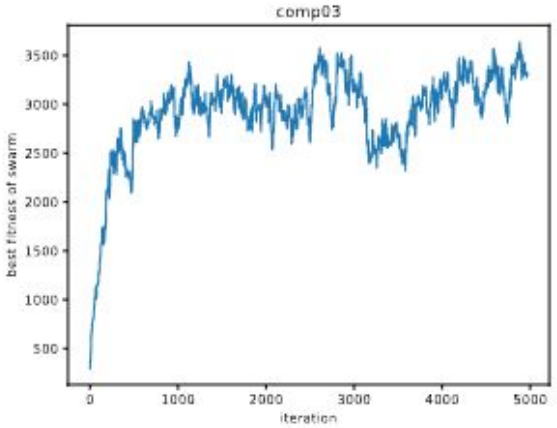
PSO ao longo das iterações (comp01-03)



(a) Instância comp01

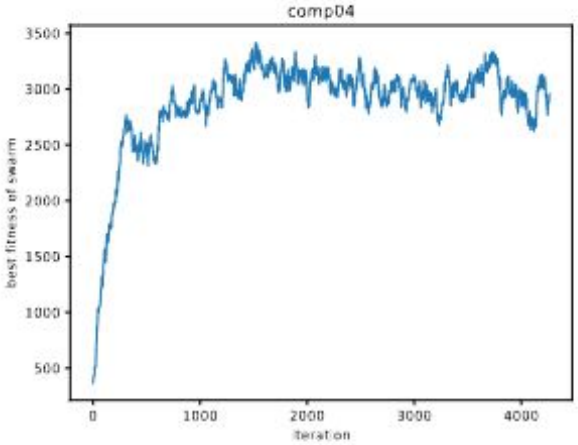


(b) Instância comp02

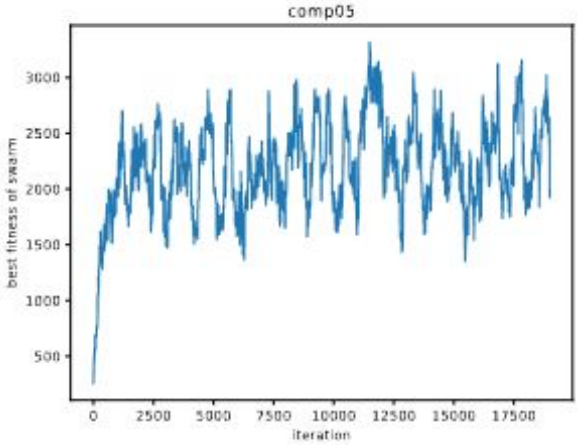


(c) Instância comp03

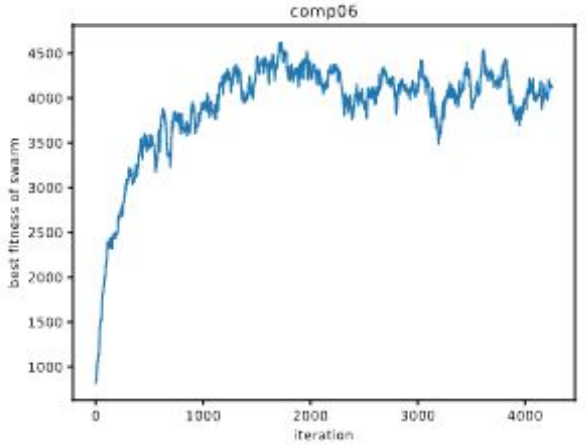
PSO ao longo das iterações (comp04-06)



(d) Instância comp04

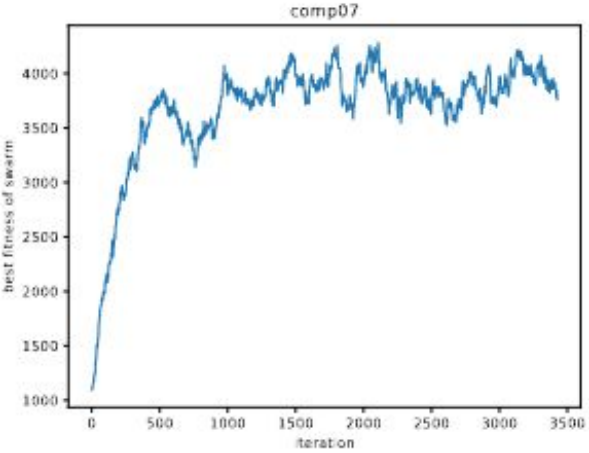


(e) Instância comp05

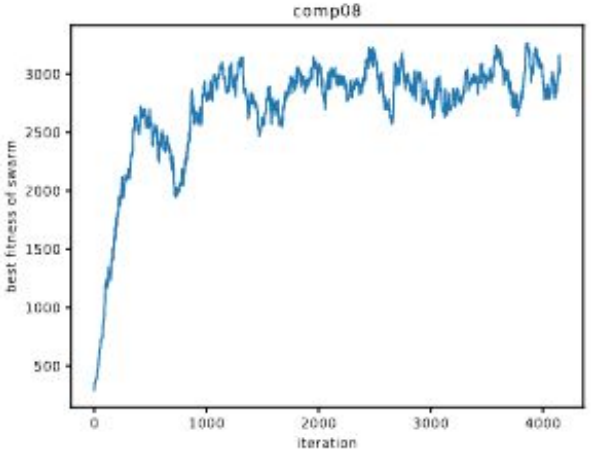


(f) Instância comp06

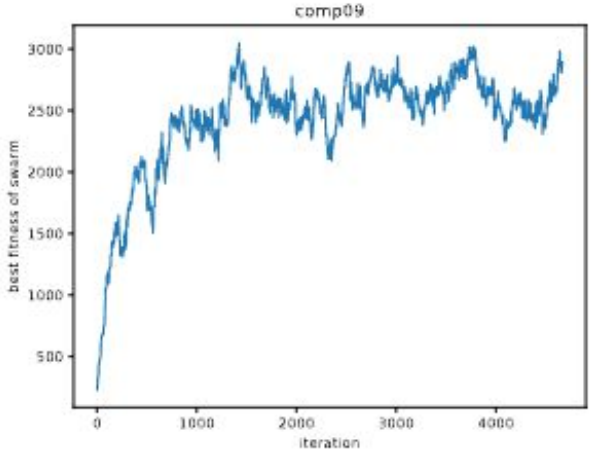
PSO ao longo das iterações (comp07-09)



(g) Instância comp07

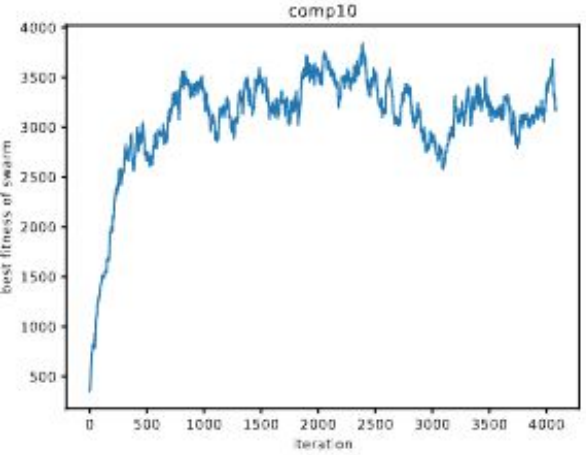


(h) Instância comp08

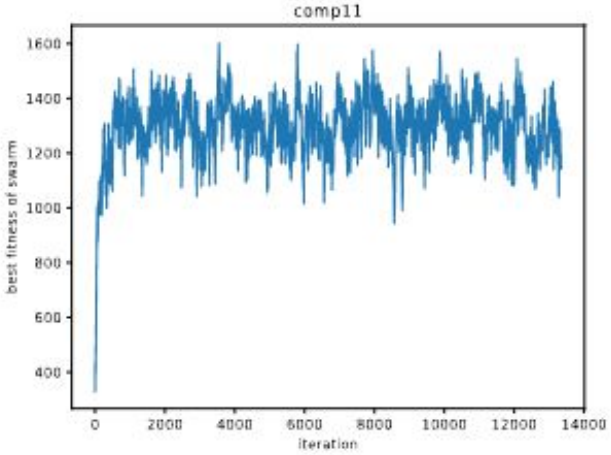


(i) Instância comp09

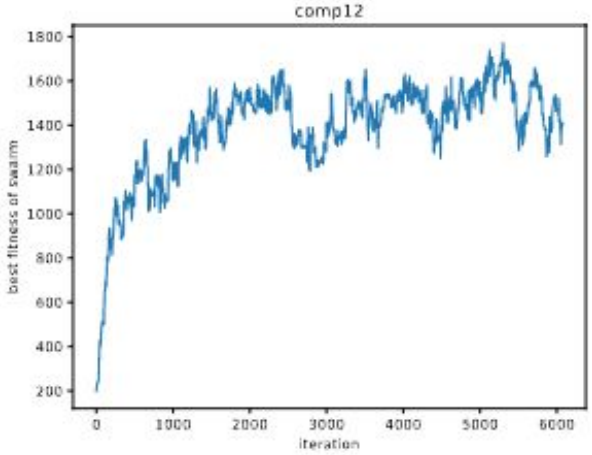
PSO ao longo das iterações (comp10-12)



(j) Instância comp10

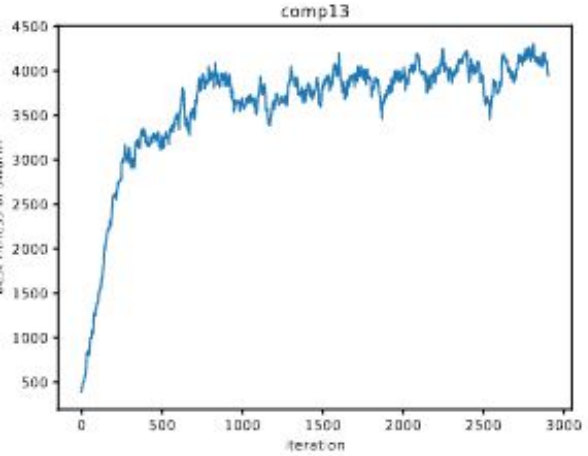


(k) Instância comp11

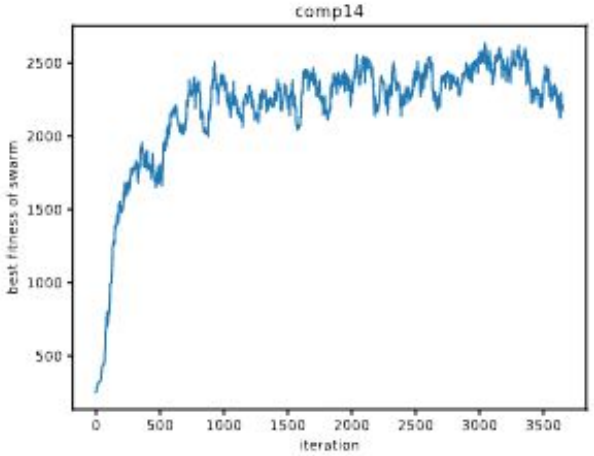


(l) Instância comp12

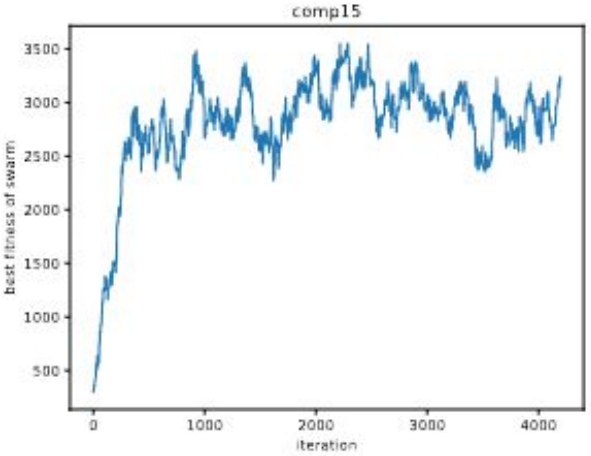
PSO ao longo das iterações (comp13-15)



(m) Instância comp13

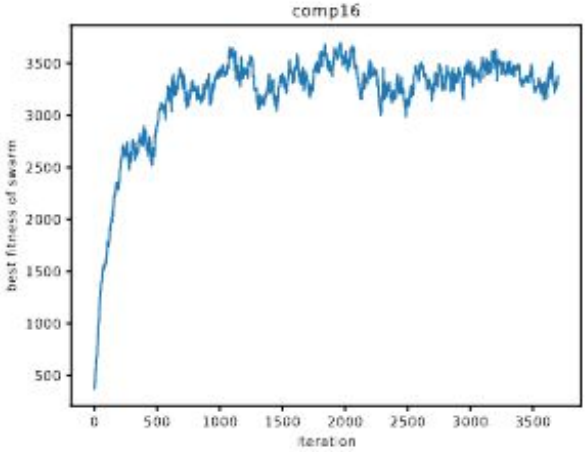


(n) Instância comp14

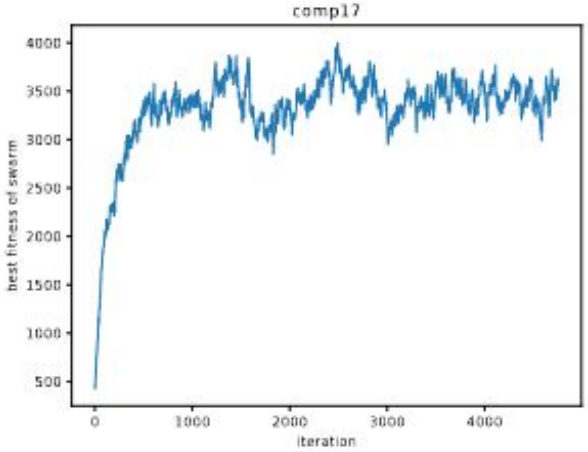


(o) Instância comp15

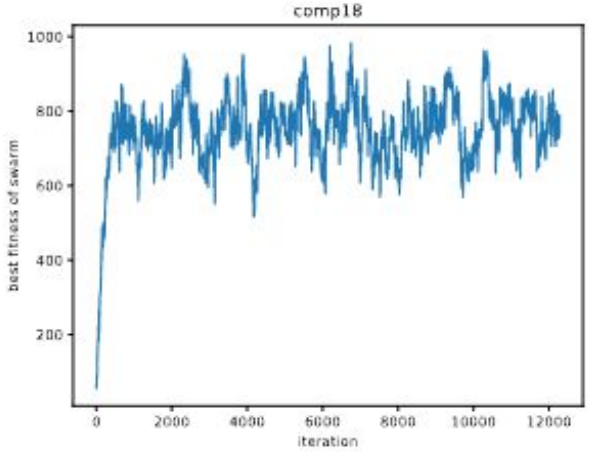
PSO ao longo das iterações (comp16-18)



(p) Instância comp16

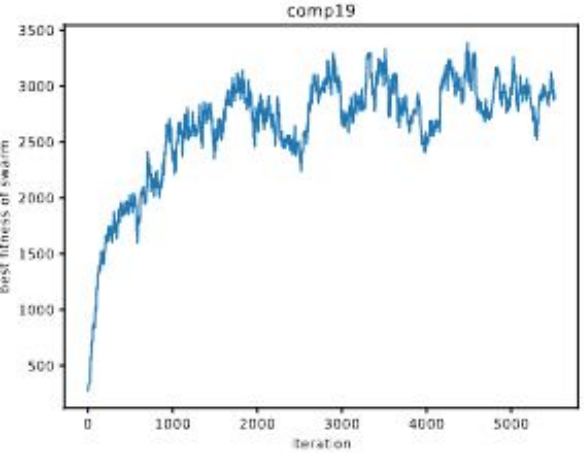


(q) Instância comp17

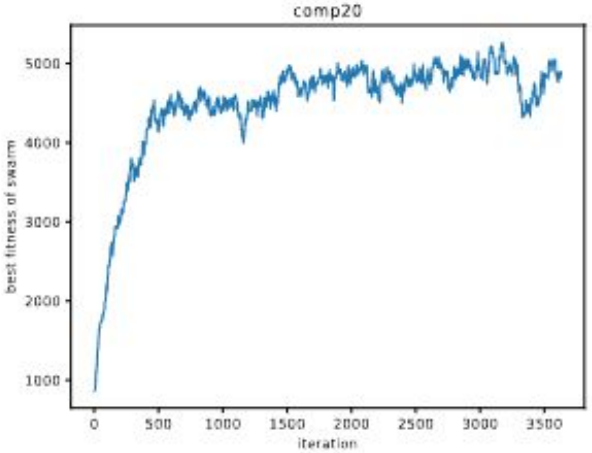


(r) Instância comp18

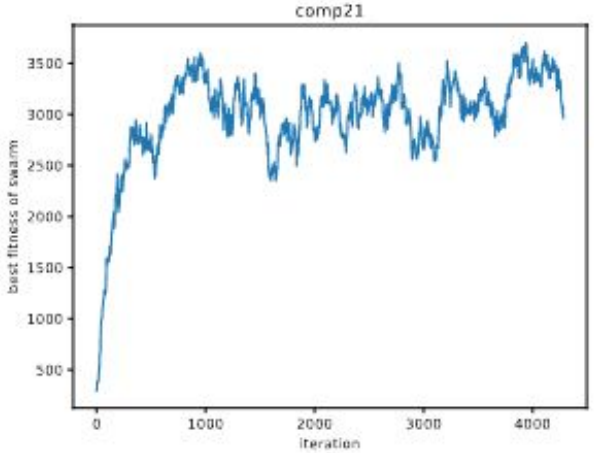
PSO ao longo das iterações (comp19-21)



(s) Instância comp19



(t) Instância comp20



(u) Instância comp21

Análise dos Resultados

- Resultados finais bons;

- Algoritmo construtivo do GRASP se
- adaptou bem ao PHTU;

- PSO performou mal;
- Rand Mutate e Rand Change não conseguiam melhorar a solução;

O que deu errado?

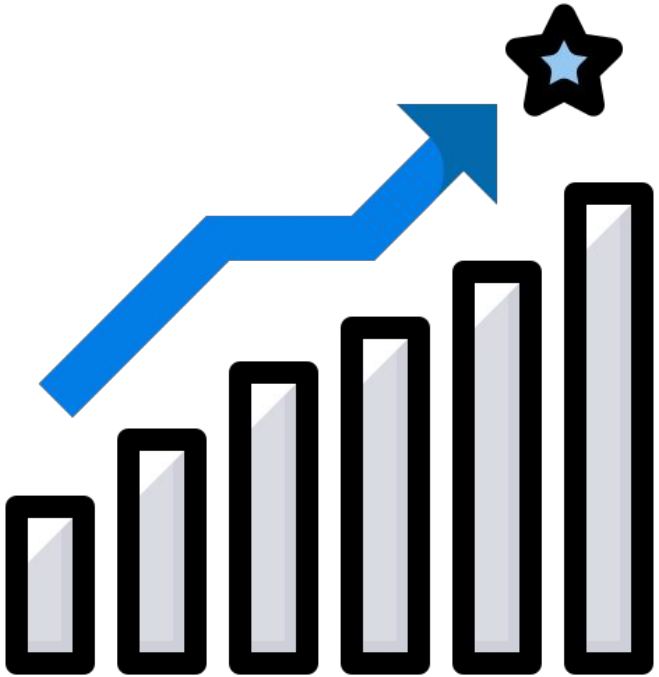
- O artigo de referência trabalhava com uma versão mais simples do PHTU;
- Mesmo forçando somente melhoras não diminuiu os resultados;

- Forçar a viabilidade atrapalhou a diversificação?
- Swaps/moves não são bons?

Possíveis melhoras

- Hibridizar o algoritmo e adotar um critério de aceitação similar ao do Simulated Annealing (SA);

- Elaborar formas de destruir e reconstruir a solução com movimentos que envolvessem mais slots, similar ao que o Large Neighborhood Search faz (LNS);



Código Fonte

https://github.com/gomesfilipe/metaheuristicas/tree/main/t2-titmetable

Referências

Chu, S.-C., Chen, Y.-T., and Ho, J.-H. (2006). Timetable scheduling using particle swarm optimization. In *First International Conference on Innovative Computing, Information and Control - Volume I (ICICIC'06)*, volume 3, pages 324–327.

de Almeida Segatto, E. (2017). Um estudo de estruturas de vizinhanças no grasp aplicado ao problema de tabela-horário para universidadess. Mestrado em ciência da computação, Universidade Federal do Espírito Santo, Vitória - ES.

Freitas, D., Lopes, L. G., and Morgado-Dias, F. (2020). Particle swarm optimisation: A historical review up to the current developments. *Entropy*, 22(3).

Kampke, E. H. (2020). *Novas Estratégias para Obtenção de Limitantes Inferiores para o Problema de Tabela-Horário de Universidades*. Doutorado em informática, Universidade Federal do Espírito Santo, Vitória - ES.

Kennedy, J. and Eberhart, R. (1995). Particle swarm optimization. In *Proceedings of ICNN'95 - International Conference on Neural Networks*, volume 4, pages 1942–1948 vol.4.

OBRIGADO!

Filipe Gomes Arante de Souza

filipe.ga.souza@edu.ufes.br