



GOVERNO DO ESTADO DO RIO DE JANEIRO  
SECRETARIA DE ESTADO DE CIÊNCIA E TECNOLOGIA  
FUNDAÇÃO DE APOIO À ESCOLA TÉCNICA  
CENTRO DE EDUCAÇÃO TECNOLÓGICA DO ESTADO DO RIO DE  
JANEIRO  
FACULDADE DE EDUCAÇÃO TECNOLÓGICA DO ESTADO DO RIO  
DE JANEIRO  
FAETERJ/PETRÓPOLIS

# Desenvolvimento e uso do Framework MVC - Lotus-PHP

Guilherme Peixoto da Costa Louro

PETRÓPOLIS  
Julho de 2015

# Desenvolvimento e uso do Framework MVC - Lotus-PHP

Guilherme Peixoto da Costa Louro

Trabalho apresentado no curso de Formação em Tecnologia da Informação e Comunicação da FAETERJ – Petrópolis como requisito parcial para obtenção do grau de tecnólogo.

Orientador: Matheus Bandini

Co-orientador: Hélio José Corrêa Barbosa

PETRÓPOLIS  
Julho de 2015

Monografia de Projeto Final de Graduação sob o título “*Desenvolvimento e uso do Framework MVC - Lotus-PHP*”, defendida por Guilherme Peixoto da Costa Louro e aprovada em Julho de 2015, em Petrópolis, Estado do Rio de Janeiro, pela banca examinadora constituída pelos professores:

---

Orientador

---

Co-orientador

---

Nome do membro da banca  
Intituição do Membro

---

Nome do membro da banca  
Intituição do Membro

# *Resumo*

Atualmente o uso de meta-heurísticas vem sendo utilizada por sua simplicidade de implementação e por abranger uma vasta gama de problemas com os mais diversos níveis de complexidade. Neste trabalho estudamos a técnica de otimização denominada Otimização por Enxame de Partículas, Particle Swarm Optimization (PSO). A qual, assim como ocorre em grande parte das meta-heurísticas, a qualidade de aplicação dessa técnica deve definir alguns parâmetros que influenciam diretamente no desempenho do algoritmo, como a inércia que impede que uma partícula mude de direção instantaneamente, e a capacidade de aprendizado coletivo e individual da partícula. Esses parâmetros foram avaliados por um conjunto de testes amplamente utilizado na literatura e com o objetivo de observar como o algoritmo se comporta com cada grupo de parâmetro em diferentes situações a ele apresentado.

Além disso, um estudo mais amplo a respeito da técnica foi feito avaliando algoritmos que adaptam seus parâmetros no decorrer da execução, buscando assim uma ferramenta mais eficaz aplicada em diferentes classes de problemas. Algoritmos que adaptam a inércia por expemplo, geralmente permitem que a partícula permaneça mais livre no início da execução do programa e tornam-se mais restritivas com o decorrer da busca.

As partículas por sua vez, não dependem apenas da inércia para evoluírem. Seu aprendizado se da por dois parâmetros conhecidos como aceleração cognitiva e aceleração social, que dizem o quanto uma partícula é influenciada para seguir o aprendizado do enxame e o quanto ela deve buscar por si própria o ponto ótimo. (Sendo assim, se uma execução mais social, ou seja, que dê mais importancia para o aprendizado do enxame, pode levar ao fracasso de todo o grupo caso ele se perca.) Com isso, fica claro que a definição desses parâmetros compromete o resultado final da técnica.

Além desses tipos de adaptação, existem algoritmos que atualizam a velocidade das partículas de maneiras diferentes de acordo com o que as partículas aprenderam, bem como a inclusão de novos parâmetros necessários ao controle da adaptação.

Portanto, este trabalho visa avaliar diversas propostas em torno da definição dos parâmetros do PSO em um conjunto de problemas de otimização sem restrições, visando identificar as propostas mais adequadas para diferentes classes de problemas.

Este trabalho, além de avaliar os parâmetros do PSO, ainda faz o uso do PSO como mais uma metaheurística em uma técnica de otimização híbrida, que procura aperfeiçoar o processo de otimização fazendo o uso de mais de uma técnica. Pelo fato da busca ser demorada utilizamos uma técnica de paralelização para diminuir o tempo de processo.

# *Abstract*

Nowadays the meta heuristics have been used for its simply of implementation and for be able to be applied in a very large range of problems with the most levels of complexity. In this work, we have study the technique named Particle Swarm Optimization (PSO). Which one, as occurs in the most of meta heuristics, the quality of application of this technique have to define some parameters that have direct influence in the performance of algorithm, like the inertia weight that prevent that the particle change its direction instatly, and the swarm learning ability and its own. Those parameters was evaluted by a group of tests largely used on literature and with the goal of observe how the algorithm behave in each group of parameters in diferents situations applied.

Furthermore, a larger study about the technique has been done evaluting algorithms that adapt their parameters during the execution, thus searching a tool more effective applied on diferents groups of problems. Algorithms that adapt the inertia weight, for instance, generally allow that the particle keep more free in the opening execution e became more restrective during the search.

Particles on its turn, do not depend olny of the inertia wight to evolve. Its learning is give by two parameters known as cognitive aceleration and social aceleration, which one, define how much the particle is influenced by the swarm and how much she must search the best by itself. (Thus, if implement more social, in other words, that give more importance to the swarm learn, it can lead to failure of all group in case it is lost.) So, it is clear that the definition of those parameters commits the final result of the technique.

Beyond those types of adaptation, still exist algorithms that updates in diferents ways the particle's velocity, according with what those particles have learn, as well as inclusion of new necessary parameters to control the adaptation.

There fore, thiw work aims to evalute various proposals around the definition of PSO parameters in a group of optimization problems without restriction, aiming identify proposals more adequate for diferents classes of problems.

# *Dedicatória*

Dedico este trabalho a

# *Agradecimentos*

Agradeço a ...



# *Sumário*

## **Lista de Figuras**

## **Lista de Tabelas**

<b>1</b>	<b>Introdução</b>	p. 11
<b>2</b>	<b>Otimização</b>	p. 12
2.1	Otimização . . . . .	p. 12
2.2	Máximos e Mínimos . . . . .	p. 13
<b>3</b>	<b>Metaheurística</b>	p. 14
3.1	Heurística e Meta-Heurística . . . . .	p. 14
3.2	Problemas e Vantagens da Meta-Heurística . . . . .	p. 15
<b>4</b>	<b>Otimização Por Enxame de Partículas</b>	p. 16
4.1	<i>Particle Swarm Optimization</i> . . . . .	p. 16
4.2	<i>Global Best Particle Swarm Optimization</i> . . . . .	p. 17
4.3	<i>Decreasing Weight Particle Swarm Optimization</i> . . . . .	p. 17
4.4	Time-Varying Acceleration Coefficients PSO . . . . .	p. 18
4.5	Fully Informed PSO . . . . .	p. 19
<b>5</b>	<b>Algoritmos e Modelo de Ilhas Paralelizadas</b>	p. 20
5.1	Modelo de Ilhas Paralelizadas . . . . .	p. 20
<b>6</b>	<b>Experimentos Computacionais</b>	p. 21

6.1 Experimentos . . . . .	p. 21
<b>7 Conclusões e trabalhos futuros</b>	p. 23
7.1 Conclusões . . . . .	p. 23
<b>Referências</b>	p. 24
<b>Anexo A – Ferramentas utilizadas</b>	p. 25

## *Lista de Figuras*

# *Lista de Tabelas*

- 1 Conjuntos de configurações para os algoritmo do PSO sem adaptação. . p. 21
- 2 Conjuntos de configurações para os algoritmo do PSO adaptativos. . . p. 22

# *1 Introdução*

Atualmente,

## 2 Otimização

Este capítulo descreve um pouco sobre otimização, o qual é o centro da base do trabalho desenvolvido.

### 2.1 Otimização

Otimização é o processo de tornar algo melhor. Consiste em tentar variações de um conceito inicial e usar informações obtidas pra melhorar esta ideia. Quando otimizamos algo, algumas questões podem surgir. A solução encontrada é a única solução? Na maioria das vezes não. É a melhor solução? Essa, é uma pergunta difícil de ser respondida. Logo, otimização é uma ferramenta da matemática que nós recorremos para termos essa resposta.

Otimização, mesmo que intuitivamente, é buscado por todos no seu cotidiano. Um exemplo disso, é a rota de sua casa até o seu trabalho. Nesse aspecto, você pode optar por otimizar tempo. Como passa todo dia pelo trajeto, sabe-se onde tem mais trânsito. Quando se evita o caminho por onde pega mais trânsito, conseqüentemente, reduz o tempo de casa até o trabalho. Ou ainda pode optar por onde tenha a menor distância, ainda que leve um tempo maior. Esse tipo de busca é conhecida como busca exaustiva.

A otimização faz parte de um ramo interdisciplinar da matemática aplicada, fazendo uso de modelos matemáticos, estatísticos e de algoritmos na ajuda da tomada de decisão. O problema em questão, pode ser de minimização ou maximização de uma dada função, a qual pode ser denominada **função alvo** ou **função objetivo**, além disso, um problema de otimização ainda existe um conjunto de **restrições**, (porém problemas desse tipo não serão analisados neste trabalho) ambos relacionados às **variáveis de decisão**.

Informalmente, otimização pode ser descrita por:

$$\text{Minimizar } f(x) \text{ sujeita a } x \in \Omega \subset \mathbb{R}^n \quad (2.1)$$

Onde a função  $f$  é a *função objetivo*, e o conjunto  $\Omega$  é o *conjunto factível*.

## 2.2 Máximos e Mínimos

## 3 *Metaheurística*

Neste capítulo será descrito brevemente sobre o que vem a ser uma heurística e metaheurística e ainda comentar sobre problemas de se utilizar uma metaheurística para problemas de otimização e as principais vantagens dessa utilização.

### 3.1 Heurística e Meta-Heurística

Primeiramente, devemos entender o que vem a ser uma heurística. Nada mais é do que uma estratégia de como resolver um determinado tipo de problema. A maneira como você irá resolver o problema em questão é conhecida como heurística. No dicionário Aurélio, temos as seguintes definições:

Do latim: *cientia heuristica* (< gr.*heuristiké* [ *téchne*], 'arte de encontrar', 'descobrir')

- Conjunto de regras e métodos que conduzem à descoberta, à invenção e à resolução de problemas.
- Procedimento pedagógico pelo qual se leva o aluno a descobrir por si mesmo a verdade que lhe querem inculcar.
- Ciência auxiliar da História, que trata da pesquisa das fontes.
- Informática: Metodologia, ou algoritmo, usado para resolver problemas por métodos que, embora não rigorosos, geralmente refletem o conhecimento humano e permitem obter uma solução satisfatória.

Já a Metaheurística, nada mais é do que uma Heurística de forma mais ampla, ou seja, ela é uma heurística “genérica”. Enquanto a Heurística é a maneira como você soluciona um problema, a Metaheurística, soluciona uma vasta gama de problemas.



## 3.2 Problemas e Vantagens da Meta-Heurística

Como já dito anteriormente, ela é aplicável a uma vasta gama de problemas, e essa é a principal vantagem de se utilizar uma metaheurística. São estratégias para resolver problemas NP - Difíceis por oferecerem melhores soluções e geralmente com um tempo de processamento menor do que por outros tipos de técnicas.

De forma geral, utilizam combinação de escolhas aleatórias, estocásticas, e conhecimento histórico dos resultados anteriormente obtidos pelo método para se guiarem e realizar suas buscas pelo espaço de pesquisa em vizinhanças dentro do espaço de pesquisa, o que evita paradas prematuras em ótimos locais.

Porém algumas dessas vantagens abrem espaço para um ponto fraco. Por não ser algo específico pra um determinado problema, é extremamente normal que ela não obtenha a melhor solução no final, claro que ela pode encontrar o ponto ótimo, mas ela não garante que seja. Contudo, os problemas de otimização, no geral, não exigem a melhor solução possível. Mesmo não sendo uma exigência, é claro, que buscamos aprimorar a ferramenta de busca para uma melhor solução final.

## 4 *Otimização Por Enxame de Partículas*

Neste capítulo, será introduzido a técnica de otimização e mostraremos a influência sofrida no algoritmo sobre os parâmetros que devem ser definidos para que a aplicação tenha um bom desempenho. Além disso, será descrito algumas adaptações da técnica inicial que tentam deixar a aplicação mais independente dos parâmetros.

### 4.1 *Particle Swarm Optimization*

Criado por Kennedy e Eberhart, a técnica de otimização conhecida como Otimização por Enxame de Partículas[Kennedy e Eberhart 1995], do ingles *Particle Swarm Optimization* (PSO), faz parte de uma grande área da computação, denominada computação evolucionista, onde o aprendizado do algoritmo ocorre no meio do processo evolutivo. O PSO é inspirado no comportamento social de um bando de pássaros na busca por alimentos. Nessa busca, os pássaros aprendem não somente com suas próprias experiências, mas também com a experiência obtida pelo bando. A técnica faz uma analogia ao bando, onde o líder do bando guia os demais pássaros, porém esse líder pode ser substituído por um outro pássaro do bando. O que vai dizer quando cada pássaro é o líder, é o que ele aprendeu. Assim, ele é conhecido como Gbest, ou seja, é a partícula que obteve melhor aptidão.

$$V_{id} = V_{id} + C_1 * rand() * (P_{id} - X_{id}) + C_2 * rand() * (P_{gd} - X_{id}) \quad (4.1)$$

$$X_{id} = X_{id} + V_{id} \quad (4.2)$$

onde,

- $V_{id}$  é a velocidade da partícula em cada dimensão,
- $P_{id}$  é a posição onde foi encontrado o Pbest,

- $P_{gd}$  é a posição onde foi encontrado o Gbest,
- $X_{id}$  é a posição atual da partícula,
- $C_1$  e  $C_2$  são as componentes cognitivas e sociais e
- $rand()$  uma função aleatória no intervalo  $[0, 1]$ .

## 4.2 Global Best Particle Swarm Optimization

O GBPSO é uma variante do PSO onde adiciona o peso da inércia a velocidade da partícula a cada iteração. Isso foi feito pois a partícula tem uma tendencia natural de se perder no meio do processo evolutivo, o que é chamado de “*explosão*”. O peso da inércia, representado por  $W$ , é uma constante que geralmente está no intervalo de  $[0.4; 0.9]$ , porém vários outros valores foram encontrados na literatura.

A inércia, é a tendencia natural de um corpo que está em movimento tem de continuar em movimento. O que faz esse corpo parar, são outras forças que agem sobre ele. Já um corpo que se encontra em repouso, tem a tendencia de continuar em repouso. Esse fenômeno, foi adicionado ao algoritmo para conter a explosão do enxame. A atualização da velocidade das partículas é feita segundo a Eq. 4.3

$$V_{id} = w * V_{id} + C_1 * rand() * (P_{id} - X_{id}) + C_2 * rand() * (P_{gd} - X_{id}) \quad (4.3)$$

## 4.3 Decreasing Weight Particle Swarm Optimization

Uma área muito estudada atualmente adapta o algoritmo no decorrer da aplicação, e com bastante campo ainda em aberto (como mostraremos no capítulo 6 e 7).

O DWPSO, é uma variante do GBPSO, variando no decorrer da aplicação o peso da inércia. A atualização da velocidade das partículas é feita segundo a Eq. 4.4, considerando o  $w_0$ , o peso determinado para a primeira iteração e o  $w_f$ , o peso determinado para a última iteração.

$$V_{id} = w_i * V_{id} + C_1 * rand() * (P_{id} - X_{id}) + C_2 * rand() * (P_{gd} - X_{id}) \quad (4.4)$$

$$w_i = w_0 - (w_0 - w_f) \frac{i}{N} \quad (4.5)$$

onde  $i$  é o número da iteração atual e  $N$  o número total de iteração a serem realizadas.

Assim, o peso da inércia diminui com o passar do tempo, fazendo com que deixe que o algoritmo se comporte e forma que explore mais o espaço de busca no início da execução, adquirindo bastante conhecimento do espaço e que mais pro final das iterações ele passe a focar mais onde ele conheceu como melhor no início.

## 4.4 Time-Varying Acceleration Coefficients PSO

O TVACPSO, é um algoritmo que adapta três parâmetros no decorrer da aplicação. Assim como o 4.3, ele adapta o peso da inércia, mas também, ajusta dois parâmetros denominados, aceleração cognitiva e aceleração social. Esses dois parâmetros são de extrema importancia para o PSO. São eles que definem o aprendizado da partícula e do enxame. A aceleração cognitiva, assim como o nome diz, cognição significa aprender, corresponde a quanto uma partícula pode aprender, ou seja, o quanto de informação que ela obtém ela vai ser capaz de “memorizar”. Já a aceleração social, está ligada a quanto o enxame como um todo é capaz de aprender.

O balanço desses dois parâmetros é muito importante pois é ele quem diz como o algoritmo vai se comportar, levando mais em conta o que a partícula aprendeu e fazer com que cada partícula utilize o seu aprendizado e um pouco do aprendizado do bando, ou se ele vai priorizar o que o bando aprendeu e dar menos importância para o individual.

Em geral, esses dois parâmetros são definidos com o valor igual a 2, porém, sua eficiência nem sempre é satisfatória. A definição de todos esses parâmetros é muito trabalhosa, e como a má definição leva a um fracasso a aplicação, o *Time-Varying Acceleration Coefficients PSO* (TVACPSO) (Ratnaweera et al., 2004), ajusta esses valores no decorrer da aplicação, deixando o algoritmo mais flexível.

$$V_{id} = w_i * V_{id} + C_{1(i)} * rand() * (P_{id} - X_{id}) + C_{2(i)} * rand() * (P_{gd} - X_{id}) \quad (4.6)$$

$$C_{1(i)} = C_{1(0)} - (C_{1(0)} - C_{1(f)}) \frac{i}{N}, \quad C_{2(i)} = C_{2(0)} - (C_{2(0)} - C_{2(f)}) \frac{i}{N} \quad (4.7)$$

## 4.5 Fully Informed PSO

O modelo denominado *Fully Informed PSO* (FIPSO) (Mendes et al., 2004) sugere que a partícula deve ser influenciada por todas as outras partículas ao seu redor. No entanto, a quantidade de vizinhos afetará positivamente, ou negativamente devido à grande variedade

de influências sofrida pela partícula. O que difere muito de outros modelos é que neles a partícula sofre influência somente da melhor partícula, descartando as demais. Sua adaptação é feita da seguinte maneira:

$$V_{id} = \chi(V_{id} + \varphi(P_m - X_{id})), \quad P_m = \frac{C_1 * P_i + C_2 * P_g}{C_1 + C_2} \quad (4.8)$$

onde  $\varphi = C_1 + C_2$ ,  $\chi$  é o fator de constrição.

## ***5 Algoritmos e Modelo de Ilhas Paralelizadas***

Aqui será descrito de maneira reduzida sobre dois algoritmos (DE e AG) utilizados no teste híbrido, além disso, ainda será explicado o paralelismo utilizado e modelo de ilhas com suas políticas de migração de indivíduos para cada ilha.

### **5.1 Modelo de Ilhas Paralelizadas**

Os algoritmos evolucionistas, possuem uma característica muito interessante de serem naturalmente paralelizáveis pelo fato das avaliações ...

## 6 Experimentos Computacionais

Nesta parte do trabalho, foram implementados os 5 algoritmos do PSO descritos anteriormente para então rodar sobre um benchmarck elaborado para testes de desempenho de algoritmos evolucionista (CEC 2005) considerando os problemas de otimização sem restrição (caso de 10 dimensões).

### 6.1 Experimentos

Primeiramente, foram avaliados cada algoritmo individualmente sobre 6 conjuntos de parâmetros diferentes (tabelas 1 e 2). Dentre esses foram selecionadas as melhores configurações, ou seja, as definições de parâmetro que chegou mais próximo do ótimo conhecido nos 25 problemas.

Tabela 1: Conjuntos de configurações para os algoritmo do PSO sem adaptação.

(a) Configurações para o PSO				(b) Configurações para o GBPSO			
PSO				GBPSO			
	C1	C2	W		C1	C2	W
Conj1	2.0000	2.0000	0.9000	Conj1	2.0000	2.0000	0.9000
Conj2	-0.2746	4.8976	0.4000	Conj2	-0.2746	4.8976	-0.3488
Conj3	-0.2746	4.8976	0.9000	Conj3	-0.2746	4.8976	0.9000
Conj4	-0.2699	3.3950	0.9000	Conj4	-0.2699	3.3950	-0.4438
Conj5	-0.6485	2.6475	0.9000	Conj5	-0.6485	2.6475	-0.6031
Conj6	-0.1565	3.8876	0.9000	Conj6	-0.1565	3.8876	-0.2256

Logo após, com os melhores de cada, foram analisados comparando não mais o algoritmo individualmente, e sim, cada algoritmo com os demais para então saber qual deles se sobre saiu nos mesmos 25 problemas.

Tabela 2: Conjuntos de configurações para os algoritmo do PSO adaptativos.

(a) Configurações para o DWPSO						
DWPSO						
	C1	C2	Wi	Wf		
Conj1	2.0000	2.0000	0.9000	0.4000		
Conj2	-0.2746	4.8976	0.2000	-0.3488		
Conj3	-0.2746	4.8976	-0.3488	0.2000		
Conj4	-0.2699	3.3950	0.0001	-0.4438		
Conj5	-0.6485	2.6475	-0.1031	-0.6031		
Conj6	-0.1565	3.8876	0.3000	-0.2256		

  

(b) Configurações para o TVACPSO						
TVACPSO						
	C1i	C1f	C2i	C2f	Wi	Wf
Conj1	2.0000	0.1000	0.1000	2.0000	0.9000	0.4000
Conj2	-0.2746	-2.2746	2.8976	4.8976	0.2000	-0.3488
Conj3	-0.2746	-2.2746	2.8976	4.8976	0.4000	0.9000
Conj4	-0.2699	-2.2699	1.3950	3.3950	0.1000	-0.4438
Conj5	-0.6485	-2.6485	0.6475	2.6475	-0.1031	-0.6031
Conj6	-0.0787	-2.0787	1.7637	3.7637	0.4787	-0.0787



## *7 Conclusões e trabalhos futuros*

### 7.1 Conclusões

## *Referências*

[Kennedy e Eberhart 1995]KENNEDY, J.; EBERHART, R. Particle swarm optimization.  
In: IEEE. *Neural Networks, 1995. Proceedings., IEEE International Conference on*. [S.l.],  
1995. v. 4, p. 1942–1948.

## *ANEXO A – Ferramentas utilizadas*

Foi feita uma análise de algumas ferramentas...