Sistemas Inteligentes Híbridos

Eduardo José M. V. dos Santos – ejmvs@cin.ufpe.br
Renato Moura Dantas – rmd2@cin.ufpe.br
Karina Mota Silva – kms3@cin.ufpe.br

Professor: Cleber Zanchettin

Apresentação

- Introdução
  - Arquitetura básica da Rede MLP
  - Objetivo
  - Definições
    - Backpropagation
    - Inteligência de enxames
    - PSO (*Particle Swarm Optimization*)
    - FSS (*Fish School Search*)
- Experimentos
- Resultados
- Gráficos
- Conclusão

Introdução (Objetivo)

- Algoritmos utilizados para treinamento
  - Backpropagation
  - Particle Swarm Optimization
  - Fish School Search
- Treinamento da Rede MLP
- Análise comparativa dos processos de treinamento

Introdução (Definições - backpropagation)

- Modo de treinamento supervisionado;
- A rede aprende um conjunto pré-definido de pares de exemplos de entrada/saída em ciclos de propagação/adaptação;

Introdução (Definições - backpropagation)

Inicialização
→ Inicializa a rede com pesos randômicos e pequenos
Treinamento (Para cada padrão do conjunto de treinamento)
→ Loop até que o erro de cada neurônio de saída seja ≤ tolerância, para todos os padrões do conjunto de treinamento ou quantidade de loops seja alcançada. (Mean Squared Error);
1. Calculam-se as saídas dos neurônios, começando da primeira camada escondida até a camada de saída;
2. Calcula-se o erro para cada neurônio da camada de saída. Se erro ≤ tolerância, para todos os neurônios, volta ao passo 1;
3. Atualizam-se os pesos de cada neurônio, começando pela camada de saída, até a primeira camada escondida;
4. Volta-se ao passo 1 caso a diferença entre a saída da rede e a resposta desejada seja maior que um determinado limite especificado pelo usuário

Introdução (Definições - Inteligência de enxames)

- Designa sistemas de inteligência artificial onde o comportamento coletivo dos indivíduos em uma população causa simples soluções coerentes ou padrões a surgir.
- Tentativa de projetar algoritmos ou dispositivos distribuídos de solução de problemas sem ter um controle centralizado, inspirado no comportamento coletivo de agentes sociais e outras sociedades animais.

Introdução (Definições - PSO)

- Inspirado no comportamento de bandos de pássaros.
- Busca por alimentos e a interação entre aves ao longo do voo são modeladas como um mecanismo de otimização;
- Área sobrevoada é equivalente ao espaço de busca e encontrar o local com comida corresponde a encontrar a solução ótima;
- O algoritmo é modelado por pássaros (chamados de partículas) que fazem uso de sua experiência e da experiência do próprio bando para encontrar a melhor região do espaço de busca.

Introdução (Definições - PSO)

Inicialização
→ Inicialmente gera-se n partículas com posições e velocidades aleatórias;
Treinamento
→ Loop até que o erro de cada partícula solução seja ≤ tolerância, para todos os padrões do conjunto de treinamento ou quantidade de loops seja alcançada. (Mean Squared Error);
1. Ajuste do Pbest de cada partícula: compara-se a melhor posição encontrada pela respectiva partícula;
2. Ajuste do Gbest: compara-se a melhor posição encontrada na população;
3. Atualiza-se a velocidade e posição a partir das equações: vi = wvi + η1.rand().(pbest - xi) + η2.rand().(gbest - xi) xi = xi + vi nas quais: • η1 (confia em si) e η2 (confia no enxame) são taxas de cognição, ou de aprendizagem social ; • pbest é a melhor posição em que a partícula ai já esteve • gbest é a melhor posição em que algum vizinho de ai já esteve. • rand() é um número aleatório não inteiro que varia de 0 a 1.
4. Volte para o Passo 1, repetir enquanto um critério pré-estabelecido não é alcançado.

PSO

- Vantagens
  - Insensível a mudança de escala das variáveis;
  - Implementação simples;
  - Adaptável a computadores paralelos;
  - Não requer cálculo de derivadas;
  - Poucos parâmetros para serem definidos pelo usuário;
  - Bom para encontrar o mínimo global;
- Desvantagens
  - Lento no ajuste fino da solução.

Introdução (FSS)

- Inspirado no comportamento de cardumes de peixes.
- Busca por alimento e a interação entre os peixes são modelados como um mecanismo de otimização;
- Área coberta pelo cardume é equivalente ao espaço de busca e encontrar o local com mais comida disponível corresponde a encontrar a solução ótima;
- O algoritmo é modelado por peixes que fazem uso de sua experiência e da experiência do próprio bando para encontrar a melhor região do espaço de busca.
- O fator evolutivo no algoritmo é caracterizado pelo peso de cada peixe.

Introdução (FSS)

- Feeding operator

$$W_i(t+1) = W_i(t) + \frac{f[x_i(t+1)] - f[x_i(t)]}{\max\{\left|f[x_i(t+1)] - f[x_i(t)]\right|\}}$$

- Collective-instintic operator

$$\overset{\rho}{x}_i(t+1) = \overset{\rho}{x}_i(t) + \frac{\sum_{i=1}^{N} \Delta \overset{\rho}{x}_{ind\ i} \{f[x_i(t+1)] - f[x_i(t)]\}}{\sum_{i=1}^{N} \{f[x_i(t+1)] - f[x_i(t)]\}}$$

Introdução (FSS)

- Collective-volitive operator

$$Bari\ (t) = \frac{\sum_{i=1}^{N} \vec{x}_i(t) W_i(t)}{\sum_{i=1}^{N} \vec{x}_i(t)}$$

$$\vec{x}_i(t+1) = \vec{x}_i(t) - step_{vol}.rand.[\vec{x}_i(t) - Bari\ (t)]$$

$$\vec{x}_i(t+1) = \vec{x}_i(t) + step_{vol}.rand.[\vec{x}_i(t) - Bari\ (t)]$$

Introdução (Definições - FSS)

Inicialização
→ Inicializa do cardume de peixes;
Treinamento
→ Loop até que o erro de cada peixe(solução) ≤ tolerância, para todos os padrões do conjunto de treinamento ou a quantidade de loops máxima seja alcançada. (Mean Squared Error);
1. Calcula-se o passo individual de cada peixe. Executa operador individual, onde cada peixe vai andar no espaço de busca para achar um lugar com comida abundante.
2. Executa operador de alimentação, onde o peso de cada peixe é atualizado de acordo com sua posição atual no espaço de busca.
3. Realiza operador coletivo de instinto. Atualiza a posição de todos os peixes de acordo com o movimento dos peixes que foram bem sucedidos no passo individual.
4. Realiza operador coletivo de controle do cardume. O cardume é dilatado ou comprimido de acordo com o peso total do mesmo. Caso o cardume esteja perdendo peso, vai ser dilatado para buscar de forma mais ampla. Caso contrário, contrai para focar a busca de forma mais local.
5. Caso não cumpra os requisitos de saída do loop, volta ao ponto 1.

Experimentos

- Experimentos executados sobre o dataset íris UCI
  - Número de instancias: 150
  - Atributos
    - sepal length in cm
    - sepal width in cm
    - petal length in cm
    - petal width in cm
  - Classe
    - Iris Setosa (1,0,0)
    - Iris Versicolour (0,1,0)
    - Iris Virginica (0,0,1)
  - Numero de Neurônios na Camada Entrada: 4
  - Numero de Neuronios na Camada Escondida: 4
  - Numero de Neurônios na Camada Saída: 3
  - Função de ativação: sigmoide

Experimentos

- Algoritmos realizam critério de parada para que meansquarederror < 0.003 ou quantidade de iterações sejam realizadas;
- Projeto desenvolvido em Java
- Disponível em:
  - https://github.com/edumarcelino/br.ufpe.cin.mlp.fss.pso/

Experimentos (configurações e cenários)

- Os algoritmos foram executados 30 vezes, com o método *10-cross-fold validation*;
- Para todos os algoritmos foram utilizados os mesmos folds nas mesmas execuções;

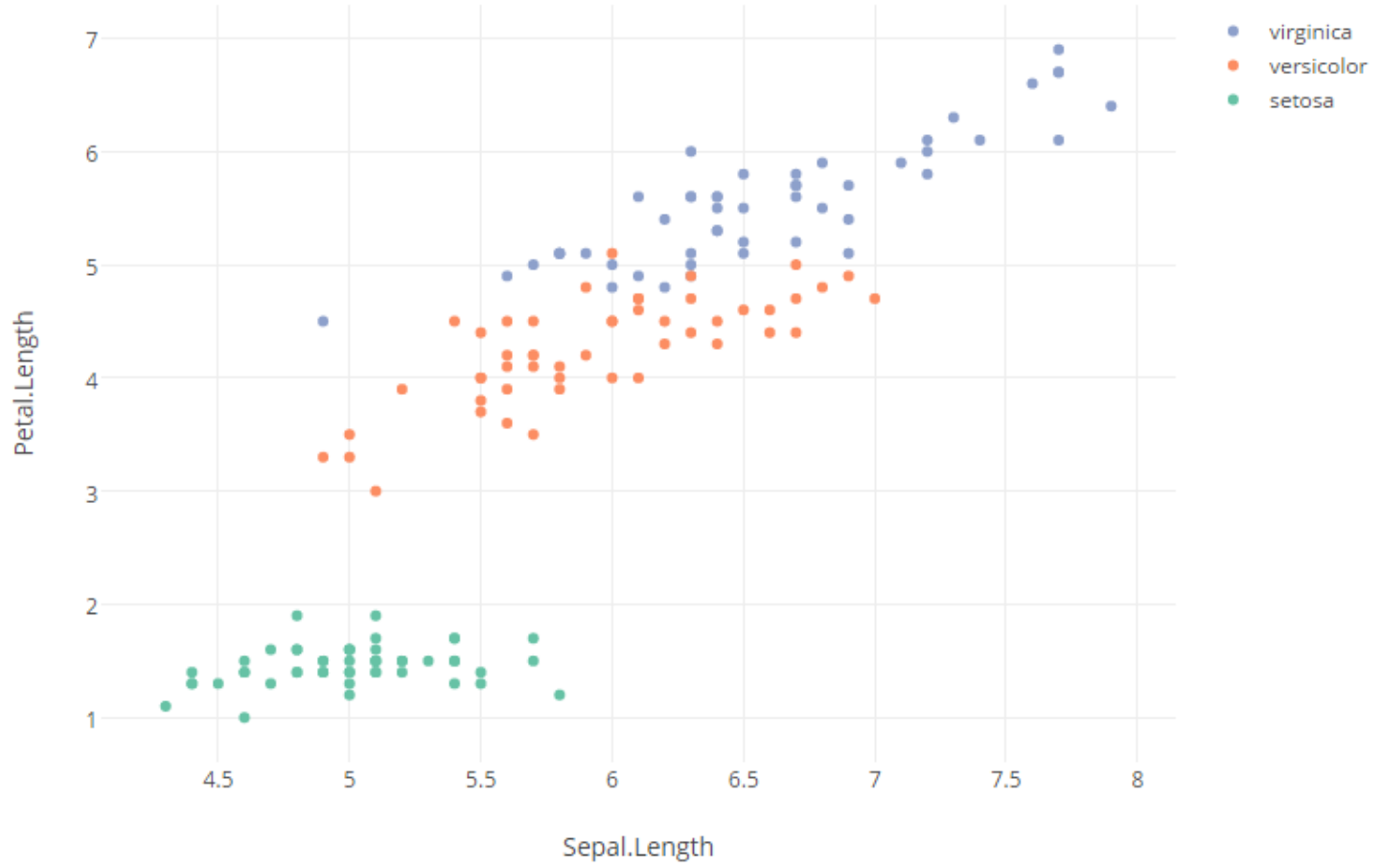
PSO

- PROB_DEATH = 0.005;
- ERRO_PARADA = 0.003;
- w = 0.729 (peso de inercia)
- c1 = 1.49445; (confia na partícula – individual)
- c2 = 1.49445 (confia no social – enxame)
- r1, r2 (randômico entre 0 e 1)

FSS

- ERRO_PARADA = 0.003;
- WEIGHT_MAX = 2500;
- WEIGHT_MIN = 100;
- *STEP_IND_INCIAL* = 10.0;
- *STEP_COLECTIVE_INCIAL* = 0.1;
- *STEP_IND_FINAL* = 1.0;
- *STEP_COLECTIVE_FINAL* = 1.0;

Experimentos (distribuição das classes)



MLP (BackPropagation) - RESULTADOS

Camada escondida: 4 neurônios
Taxa de aprendizagem: 0.1

Época	Acertos médios com intervalo de confiança 95%	Execução(ms)
1	[0.8901497 ± 0.001916482]	1280
2	[0.8977941 ± 0.001256433]	1332
3	[0.9046496 ± 0.001510149]	1399
4	[0.9097195 ± 0.0004997302]	1449
5	[0.9118233 ± 0.0007238652]	1587
10	[0.9309946 ± 0.0004073444]	1921
20	[0.9494725 ± 0.000569533]	2593
50	[0.9502167 ± 0.0004239025]	5426
100	**[0.9520389 ± 0.001410334]**	**15110**

Camada escondida: 4 neurônios
Taxa de aprendizagem: 1.0

Época	Acertos médios com intervalo de confiança 95%	Execução(ms)
1	[0.7056305 ± 0.00153062]	730
2	[0.8239816 ± 0.001116979]	755
3	[0.8897426 ± 0.002350013]	877
4	[0.9111501 ± 0.001739251]	868
5	[0.9155942 ± 0.001462063]	962
10	[0.9277734 ± 0.0008893731]	1331
20	[0.924929 ± 0.001828787]	2031
50	[0.9276526 ± 0.002409297]	4276
100	**[0.939955 ± 0.002678906]**	**8108**

MLP (BackPropagation) - RESULTADOS

Camada escondida: 4 neurônios
Taxa de aprendizagem: 0.001

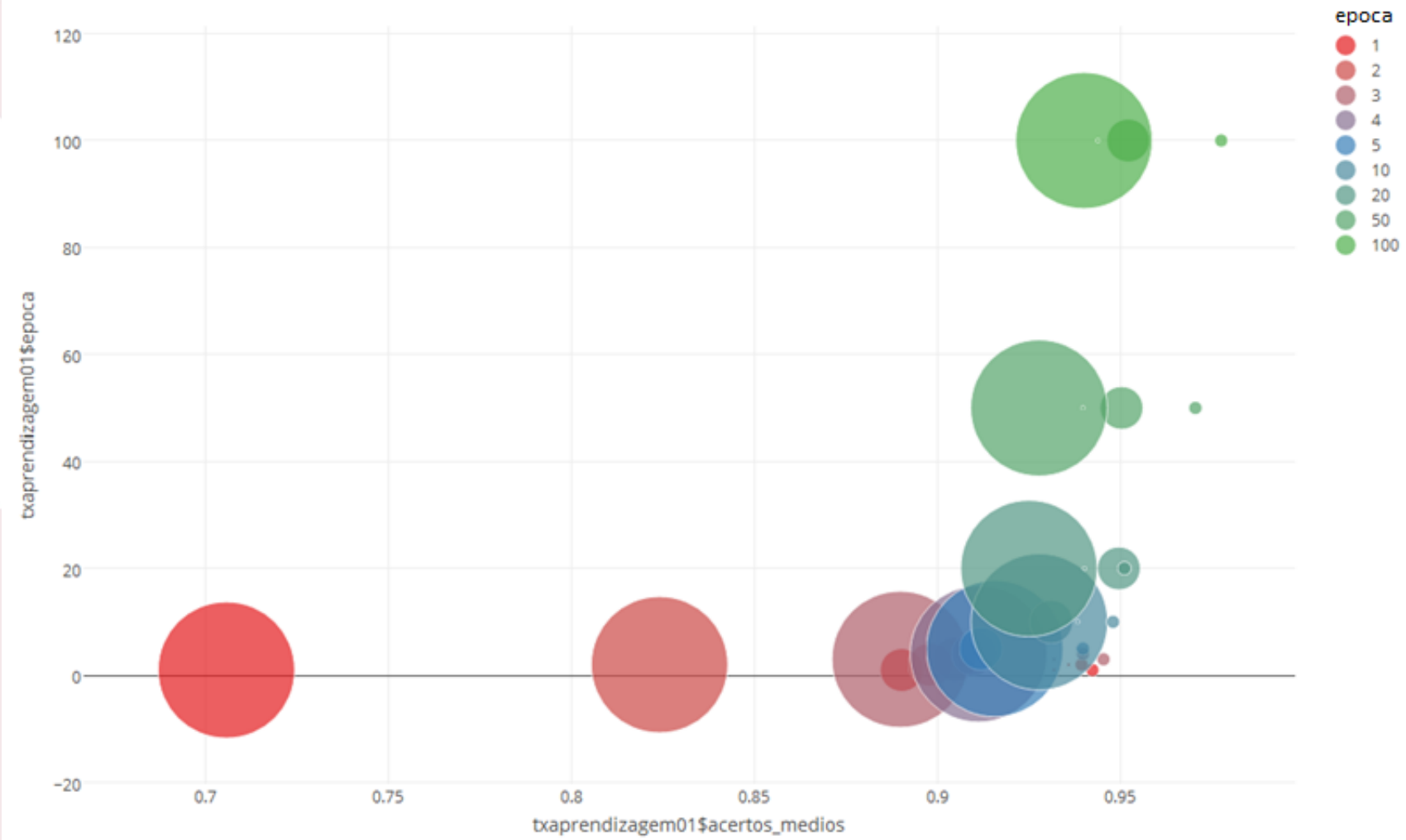
Época	Acertos médios com intervalo de confiança 95%	Execução(ms)
1	[0.9317719 ± 0.0033382]	88747
2	[0.9357378 ± 0.0031015]	85512
3	[0.9317719 ± 0.0033382]	118647
4	[0.9396237 ± 0,0026409]	93434
5	[0.9383288 ± 0.0029379]	87471
10	[0.9382478 ± 0.0022178]	96930
20	[0.9400964 ± 0.002554]	86822
50	[0.9396549 ± 0.0021157]	95875
100	**[0.9437902 ± 0,0023357]**	**93597**

Camada escondida: 4 neurônios
Taxa de aprendizagem: 0.01

Época	Acertos médios com intervalo de confiança 95%	Execução(ms)
1	[0.9422698 ± 0.002481]	10555
2	[0.9393003 ± 0.0028468]	9130
3	[0.9452869 ± 0.0025806]	9137
4	[0.939676 ± 0.0023803]	11630
5	[0.939676 ± 0.0025119]	9201
10	[0.9478994 ± 0.0028932]	9444
20	[0.9509932 ± 0.0015956]	10432
50	[0.970392 ± 0.001039785]	12221
100	**[0.9774542 ± 0.000607109]**	**16040**

MLP

(Backpropagation – tamanho tx_aprendizagem, época e acertos medios)



MLP (FSS) - RESULTADOS

Camada escondida: 4 neurônios
NUM_MAX_ITERACOES = 2000
ERRO_PARADA = 0.003

# Peixes	Acertos médios com intervalo de confiança 95%	Execução(ms)
10	[0.7400622 ± 0.002618108]	2079332
20	[0.75131 ± 0.00327451]	4065272
50	[0.8263686 ± 0.007612596]	8997304
100	**[0.8651397 ± 0.003826757]**	**18071344**

Camada escondida: 4 neurônios
NUM_MAX_ITERACOES = 1000
ERRO_PARADA = 0.003

# Peixes	Acertos médios com intervalo de confiança 95%	Execução(ms)
10	[0.6976257 ± 0.004254434]	949967
20	[0.7384432 ± 0.01146071]	2047036
50	[0.7694533 ± 0.00412913]	5065341
100	**[0.8148551 ± 0.004396531]**	**10081589**

Camada escondida: 4 neurônios
NUM_MAX_ITERACOES = 500
ERRO_PARADA = 0.003

# Peixes	Acertos médios com intervalo de confiança 95%	Execução(ms)
10	[0.6952826 ± 0.004176539]	502502
20	[0.7061519 ± 0.006418387]	972441
50	[0.7257538 ± 0.01153941]	2377666
100	**[0.7840414 ± 0.002221195]**	**4477020**

Camada escondida: 4 neurônios
NUM_MAX_ITERACOES = 100
ERRO_PARADA = 0.003

# Peixes	Acertos médios com intervalo de confiança 95%	Execução(ms)
10	[0.6425877 ± 0.006925593]	102616
20	[0.6576215 ± 0.002606142]	200753
50	[0.6804142 ± 0.003423674]	462885
100	**[0.7044392 ± 0.003033821]**	**982380**

MLP (PSO) - RESULTADOS

Camada escondida: 4 neurônios
MAX_EPOCHS = 2000
ERRO_PARADA = 0.003

Partículas	Acertos médios com intervalo de confiança 95%	Execução(ms)
10	[0.8878762 ± 0.002414567]	672780
20	[0.9353343 ± 0.003125277]	1343399
50	[0.9497194 ± 0.002405381]	3331224
100	**[0.9556076 ± 0.001786972]**	**6666725**

Camada escondida: 4 neurônios
MAX_EPOCH = 1000
ERRO_PARADA = 0.003

Partículas	Acertos médios com intervalo de confiança 95%	Execução(ms)
10	[0.8628827 ± 0.005976809]	342155
20	[0.8672212 ± 0.007283481]	665051
50	[0.9215726 ± 0.003783147]	1727129
100	**[0.9441913 ± 0.008684131]**	**3322209**

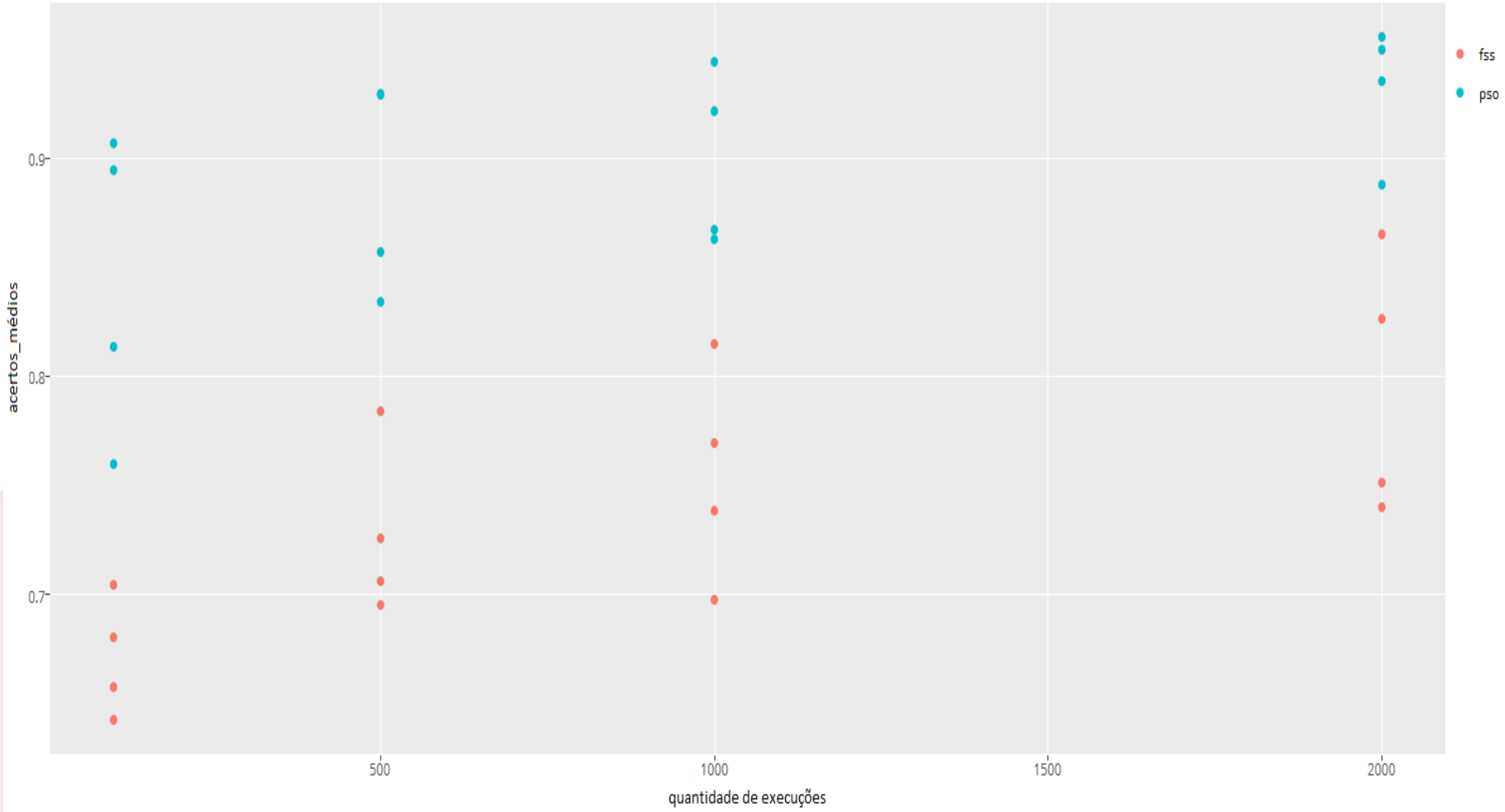
Camada escondida: 4 neurônios
MAX_EPOCH = 500
ERRO_PARADA = 0.003

Partículas	Acertos médios com intervalo de confiança 95%	Execução(ms)
10	[0.8341746 ± 0.003803305]	169333
20	[0.8570031 ± 0.008358022]	343497
50	[0.9290368 ± 0.003925289]	852360
100	**0.9295367 ± 0.007094071**	**1701115**

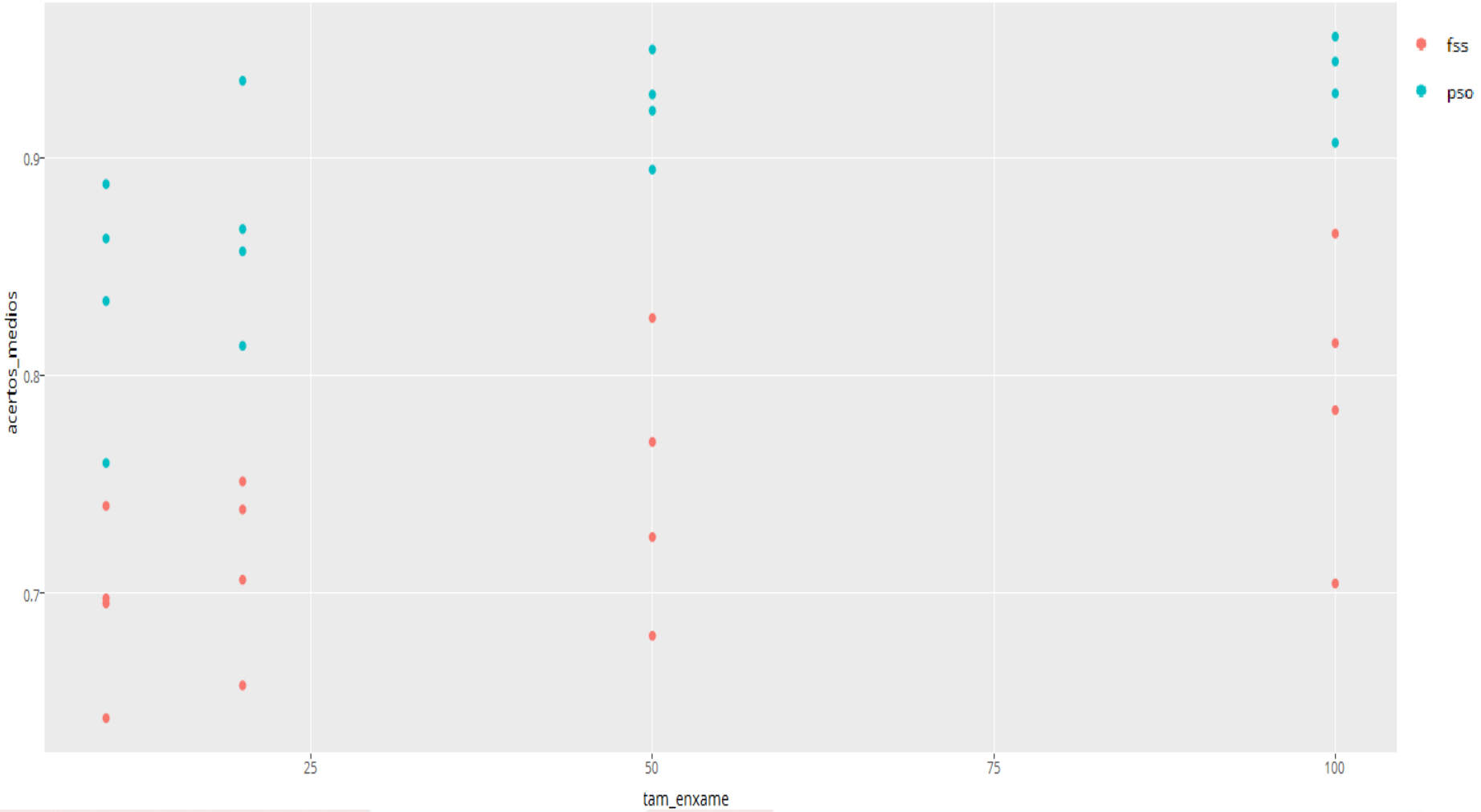
Camada escondida: 4 neurônios
MAX_EPOCH = 100
ERRO_PARADA = 0.003

Partículas	Acertos médios com intervalo de confiança 95%	Execução(ms)
10	[0.7597782 ± 0.005690167]	34683
20	[0.8135762 ± 0.004045527]	69089
50	[0.8945273 ± 0.007808492]	174336
100	**[0.9068747 ± 0.003727385]**	**346631**

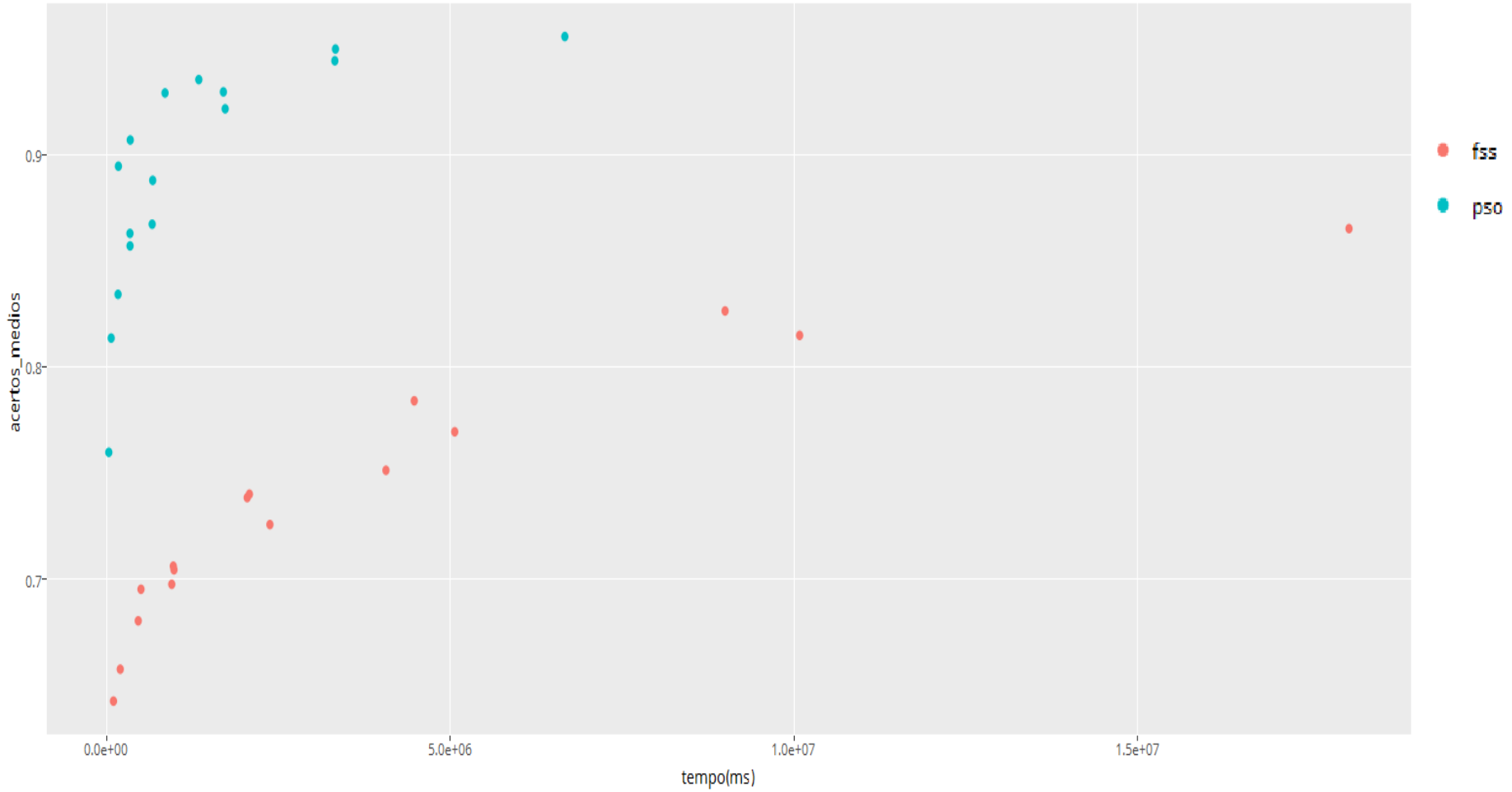
Gráficos (acertos médios vs quantidade de execuções)



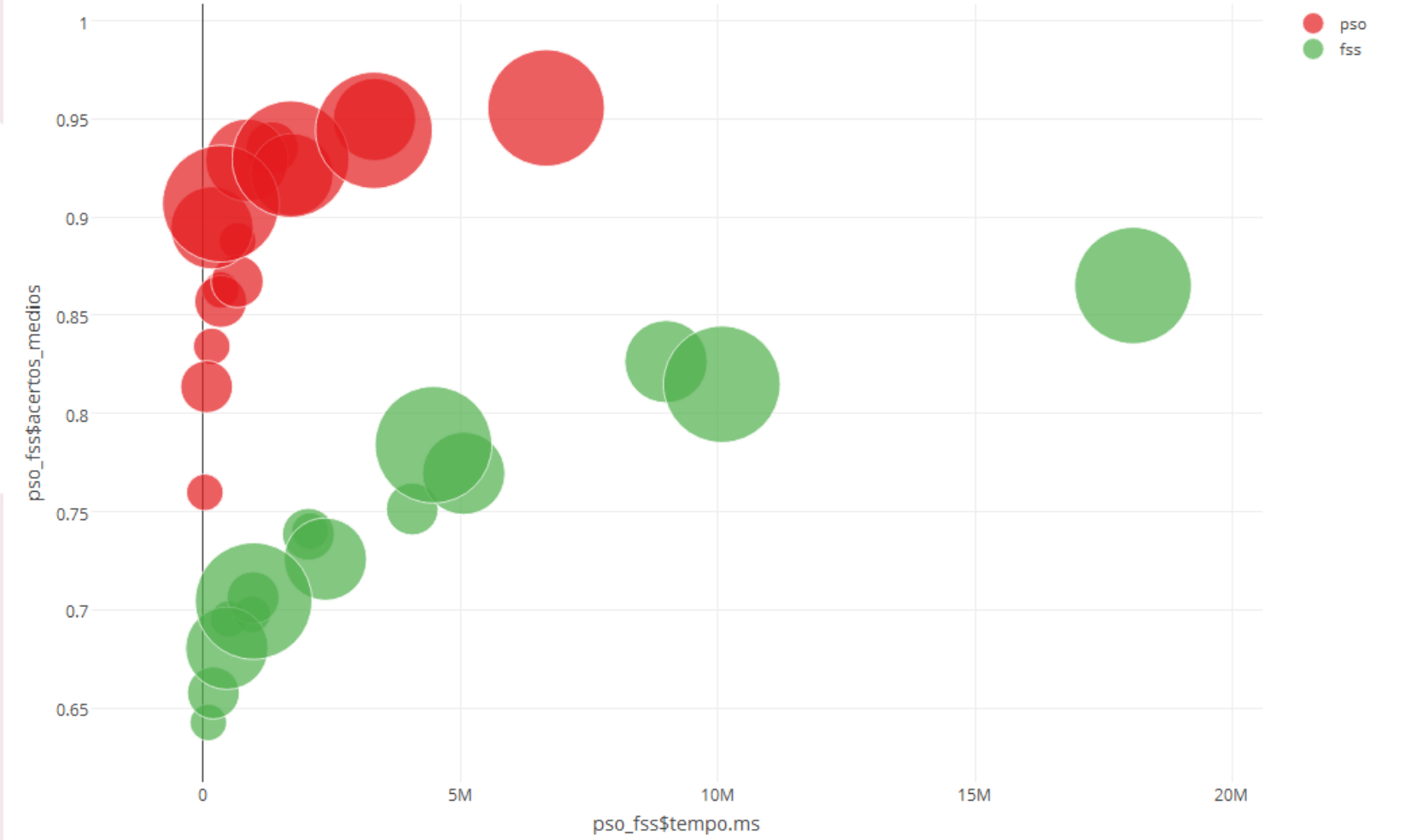
Gráficos (acertos médios vs tamanho do enxame)



Gráficos (acertos médios vs tempo de execução (ms))

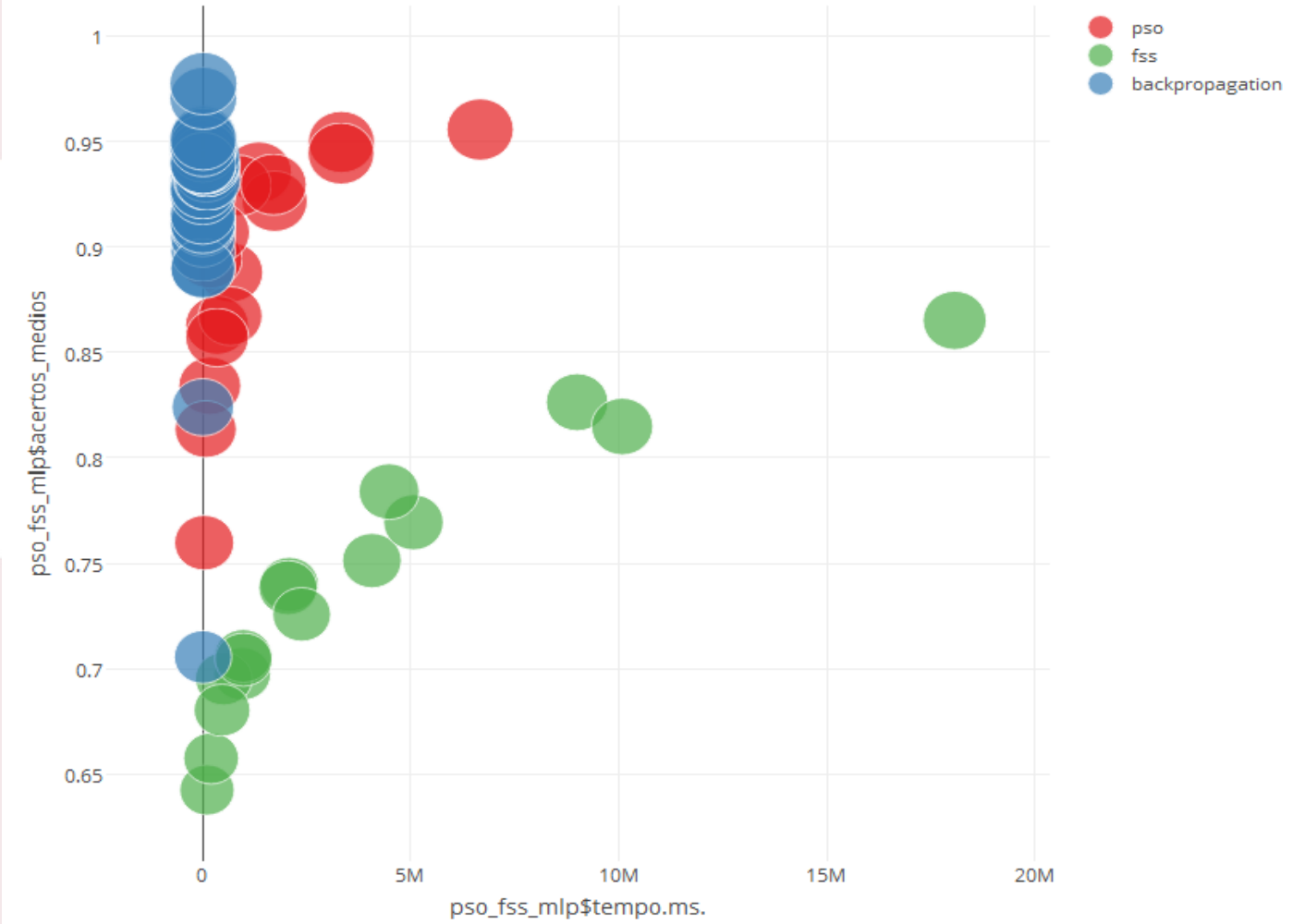


Gráficos (tamanho do enxame vs tempo vs acertos_medios)



Gráficos (tamanho do enxame vs quantidade_execucoes vs acertos_medios)

Gráficos (acertos_médios_tempo_algoritmos)



Conclusões

- Algoritmo backpropagation apresentou melhor tempo de treinamento e melhores resultados, nos cenários;
- Entre as abordagens de enxame, o algoritmo PSO apresentou menor tempo de treinamento e melhores resultados;
- O algoritmo baseado em cardumes de peixes (FSS), possui alto grau de paralelização, sendo assim, seria interessante rodar em ambiente paralelizado e com alto números de peixes;
- Da mesma forma, interessante rodar o PSO em ambiente paralelizado com alto numero de partícula;

Conclusões (trabalhos futuros)

- Realizar os experimentos em outros datasets;
- Incorporar novas técnicas de treinamento baseadas em enxame;
- Executar as versões dos algoritmos paralelizados em GPU e analisar os resultados;

Referências

- MADEIRO, Salomão S.; BASTOS-FILHO, Carmelo J. A.; LIMA NETO, Fernando B. de. "Multimodal Optimization based on Fish School Behavior" to appear in Swarm Intelligence Journal, 2012, Springer. [SUBMITTED]
- BASTOS-FILHO, Carmelo J. A.; LIMA NETO, Fernando B. de. "Fish School Search - Intelligent Algorithms for Optimization", 2011, Short-Course in XI School of Computational Intelligence (X Brazilian Congress on Computational Intelligence), Fortaleza-CE, Brazil.
- KENNEDY, James. Particle swarm optimization. In: Encyclopedia of machine learning. Springer US, 2011. p. 760-766.
- Leung, Henry, and Simon Haykin. "The complex backpropagation algorithm."IEEE Transactions on signal processing 39.9 (1991): 2101-2104.