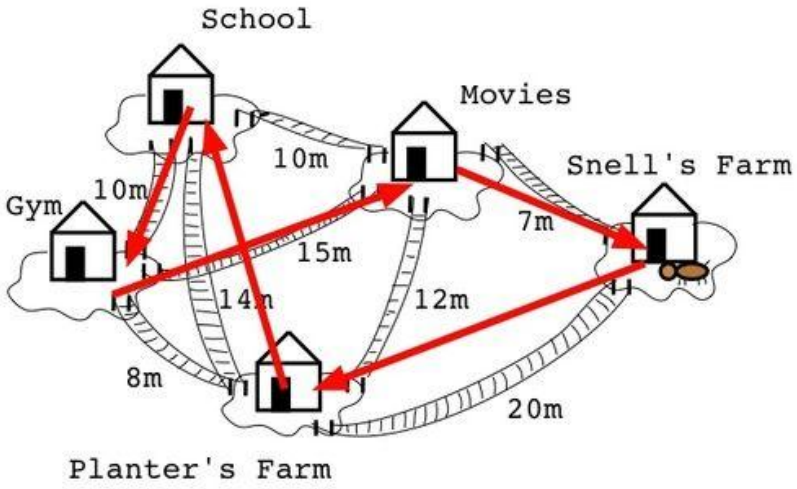
Meta-Heurística Híbridas para problemas de roteamento

Aluno:Mateus Gomes Cabana

Orientador:Profª Drª Andréa Carla Gonçalves Vianna

O Problema

- O Problema do Caixeiro Viajante (*Traveling Salesman Problem*) consiste em determinar a sequência de cidades (ou clientes) a serem visitadas por um caixeiro viajante tal que minimize a distância total percorrida e garanta que cada cidade (ou cliente) seja visitada exatamente uma vez.



Objetivos

- Estudo do Problema do Caixeiro Viajante, e sua resolução utilizando meta-heurísticas híbridas
  - Implementar computacionalmente algoritmos de meta-heurísticas e aplicá-los na resolução do Problema do Caixeiro Viajante
  - Utilização de uma rede neural para compor a meta-heurísticas, criando assim um sistema híbrido.
  - Criação de um software de roteamento com base neste sistema hibrido

REVISÃO BIBLIOGRÁFICA

- Este trabalho foi dividido nas seguintes vertentes
- Algoritmo Genético
- Busca por Vizinhança
- Busca Tabu
- Rede Neural
- Resultados
- Software

Algoritmo Genético

- Algoritmos Genéticos (AGs) é uma técnica inspirada na seleção e evolução natural de Darwin, como também outras teorias evolutivas

- “os indivíduos que apresentam características mais aptas para o ambiente tem mais chance de sobreviver e deixar seus herdeiros.”

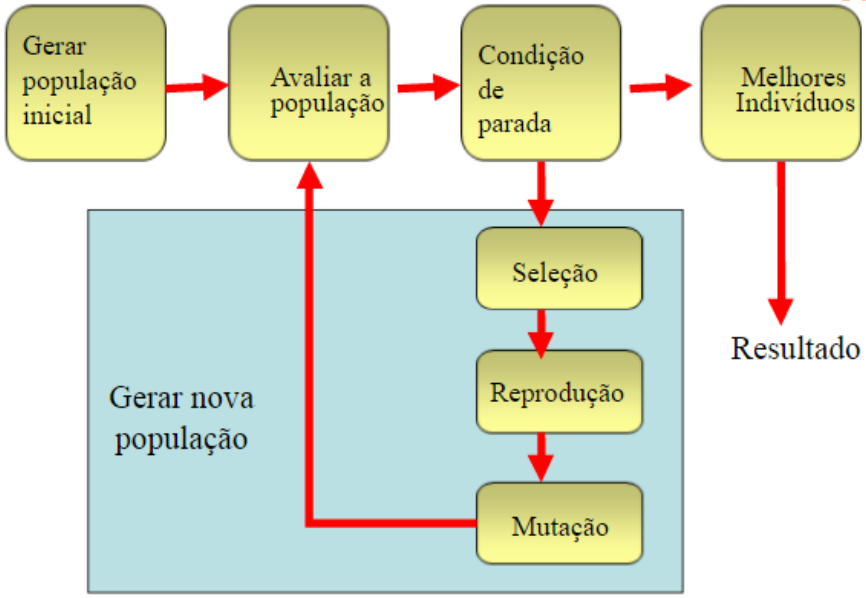
- Para compreender o funcionamento dos AGs para o Problema do Caixeiro Viajante faz-se necessário realizar uma analogia à explicação sobre a evolução das espécies. Assim, para o Problema do Caixeiro Viajante tem-se:

- **1-** Inicialmente é gerada uma população formada por um conjunto aleatório de indivíduos (possíveis rotas), que podem ser vistos como possíveis soluções para o problema.

- **2-** Durante o processo evolutivo, esta população é avaliada, sendo que para cada indivíduo (rota) é atribuída uma nota (distancia total do percurso), ou índice, que reflete sua habilidade de adaptação a determinado ambiente. Como o objetivo é minimização, as maiores notas tendem sempre a sair.

- **3-** A cada iteração os membros mantidos pela seleção podem sofrer modificações em suas características fundamentais por meio de cruzamentos (*crossover*), mutações, gerando descendentes para a próxima geração

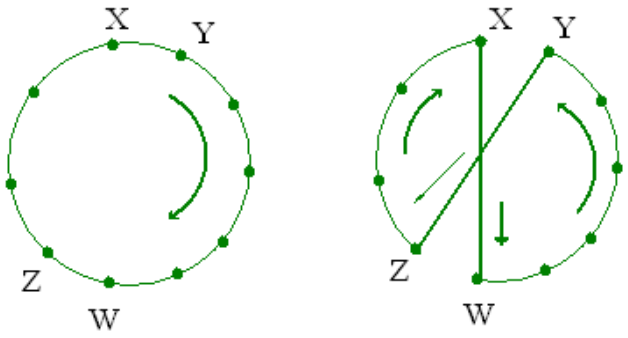
- **4-** Este processo, chamado de reprodução, é repetido até que uma solução satisfatória seja encontrada, ou o numero de iterações atinja o máximo estabelecido



Busca em Vizinhança Variavel

- O algoritmo de Busca em Vizinhança Variável (*Variable Neighbourhood Search* - VNS) é uma meta-heurística que se baseia na ideia de “mudanças sistemáticas” da vizinhança a qual a solução atual se encontra

- método VNS não segue uma trajetória, mas sim explora vizinhanças gradativamente mais “distantes” da solução corrente. O método focaliza a busca em torno de uma nova solução se e somente se um movimento de melhora é realizado.



- O Algoritmo Busca por Vizinhança consiste nos seguintes passos:
- **1** Inicialmente são criadas soluções (rotas) de forma aleatória
- **2** Seleciona-se aleatoriamente um vizinho *s'* dentro da vizinhança N(k)(*s*) da solução *s* corrente.
- **3** Esse vizinho é então submetido a um procedimento de busca local (trocas sistemáticas). Se a nova solução ótima local, *s"*, for melhor que a solução *s* corrente, a busca continua de *s"* recomeçando da primeira estrutura de vizinhança N(1)(*s*). Caso contrário, continua-se a busca a partir da próxima estrutura de vizinhança N(*k*+1)(*s*).
- **4** Este procedimento é encerrado quando uma condição de parada for atingida, no caso, o número máximo de iterações.

Busca Tabu

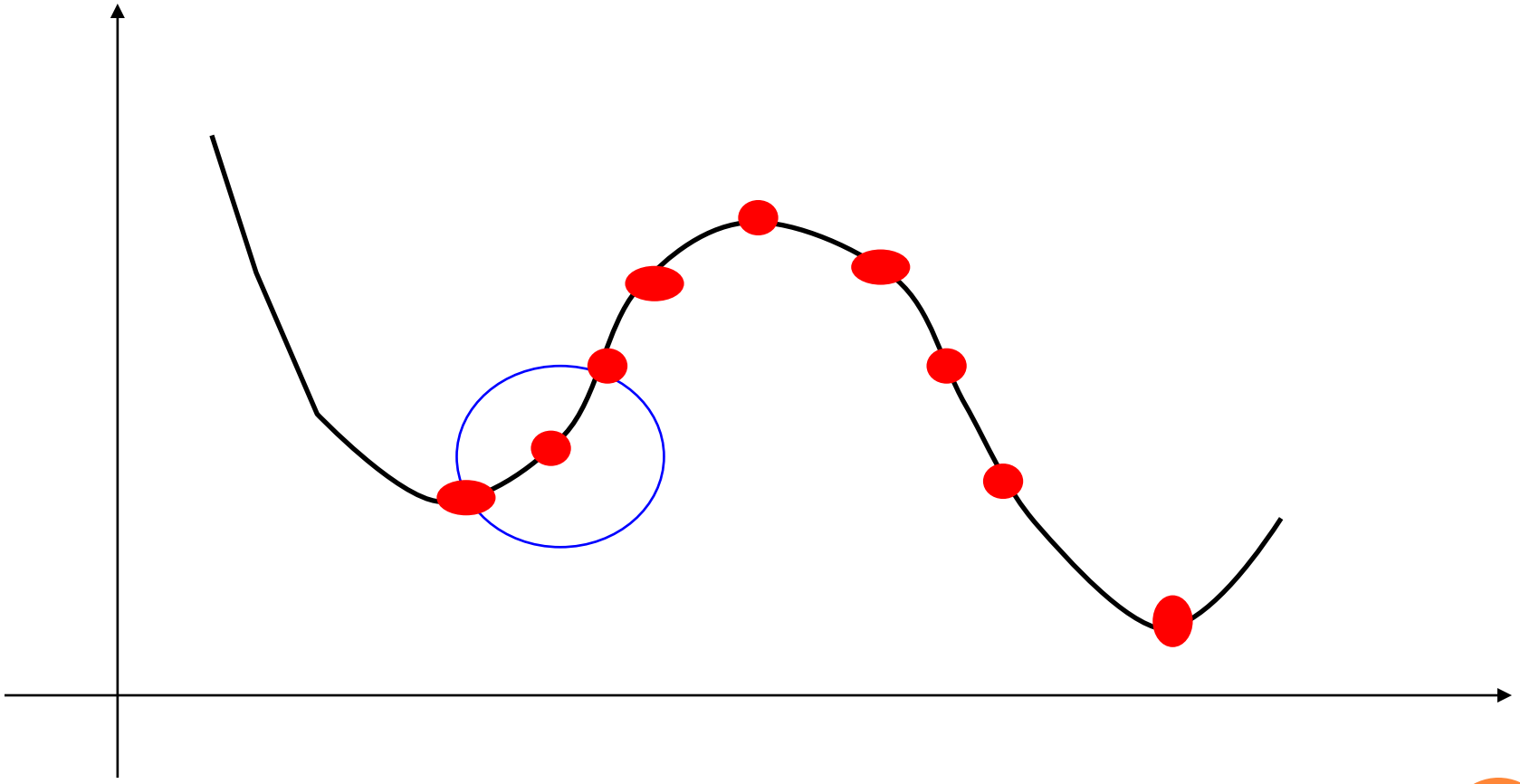
- A Busca Tabu é uma meta-heurística que tem por finalidade encontrar um melhor resultado através de uma busca heurística local (Muito semelhante ao Busca por vizinhança variável)

- A grande diferença é que o método guarda informações sobre soluções já visitadas, buscando fugir de mínimos locais, onde essa memória (lista) adaptativa vai gerar uma estratégia de busca.

Algoritmo Busca Tabu

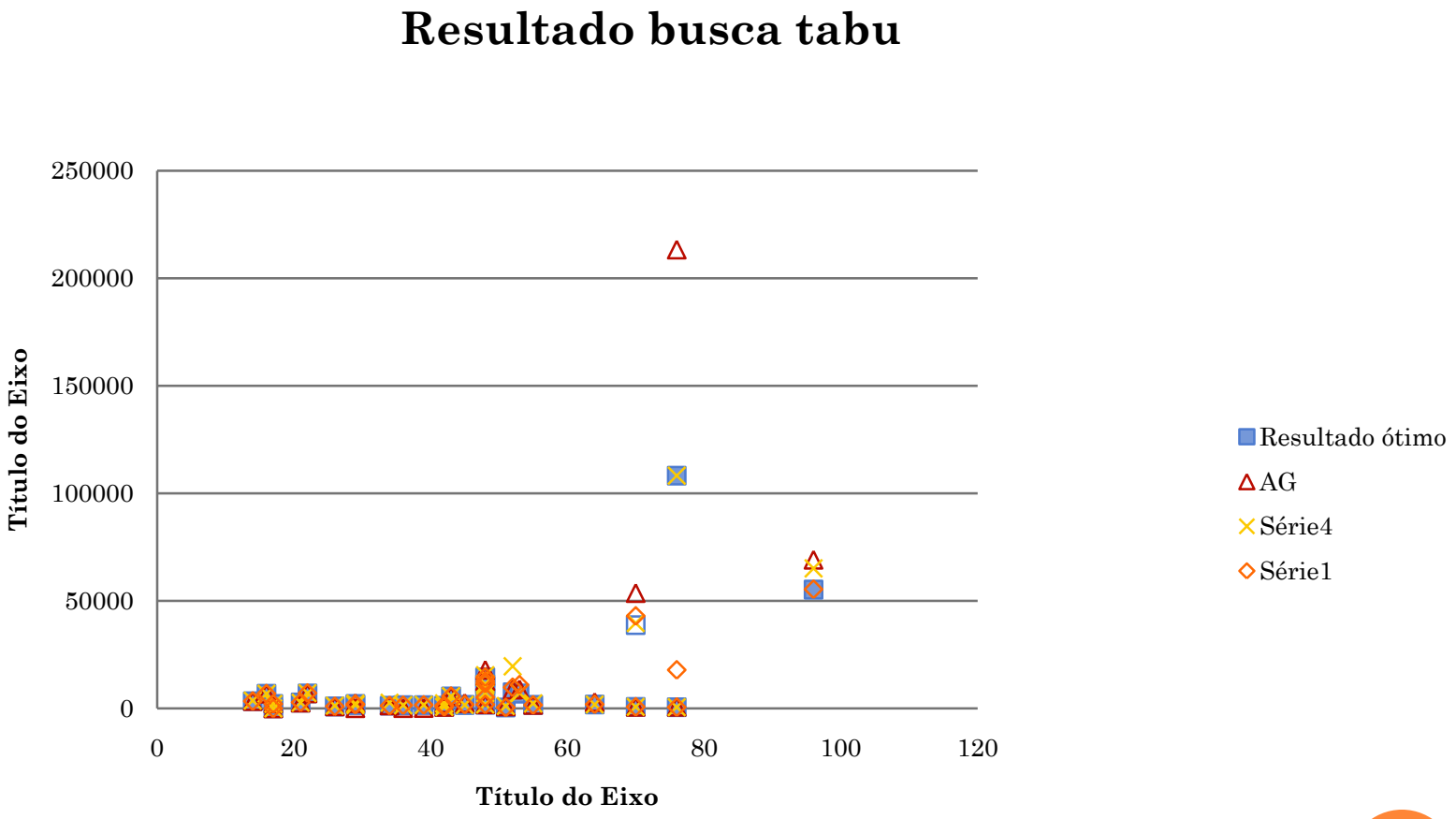
- **1-** Inicialmente são criadas soluções (rotas) de forma aleatória.
- **2-** A partir das cidades da solução da solução inicial, explora-se a vizinhança dessas cidades como no Passo 2 da Busca por Vizinhança.
- **3-** Se a solução ficar presa em uma planície à memória (lista), aumenta seu tamanho para guardar mais soluções, com o intuito de fugir da mesma. Caso isso não acontecer o programa fica com uma memória (lista) curta e de mesmo tamanho.
- **4-** A cada nova iteração é sempre consultado a lista tabu e, se possível, realiza a movimentação, senão for possível volta ao Passo 2.
- **5-** Este procedimento é encerrado quando uma condição de parada for atingida, no caso, o número máximo de iterações ou senão houver melhora na solução corrente.

Problema de Ciclagem

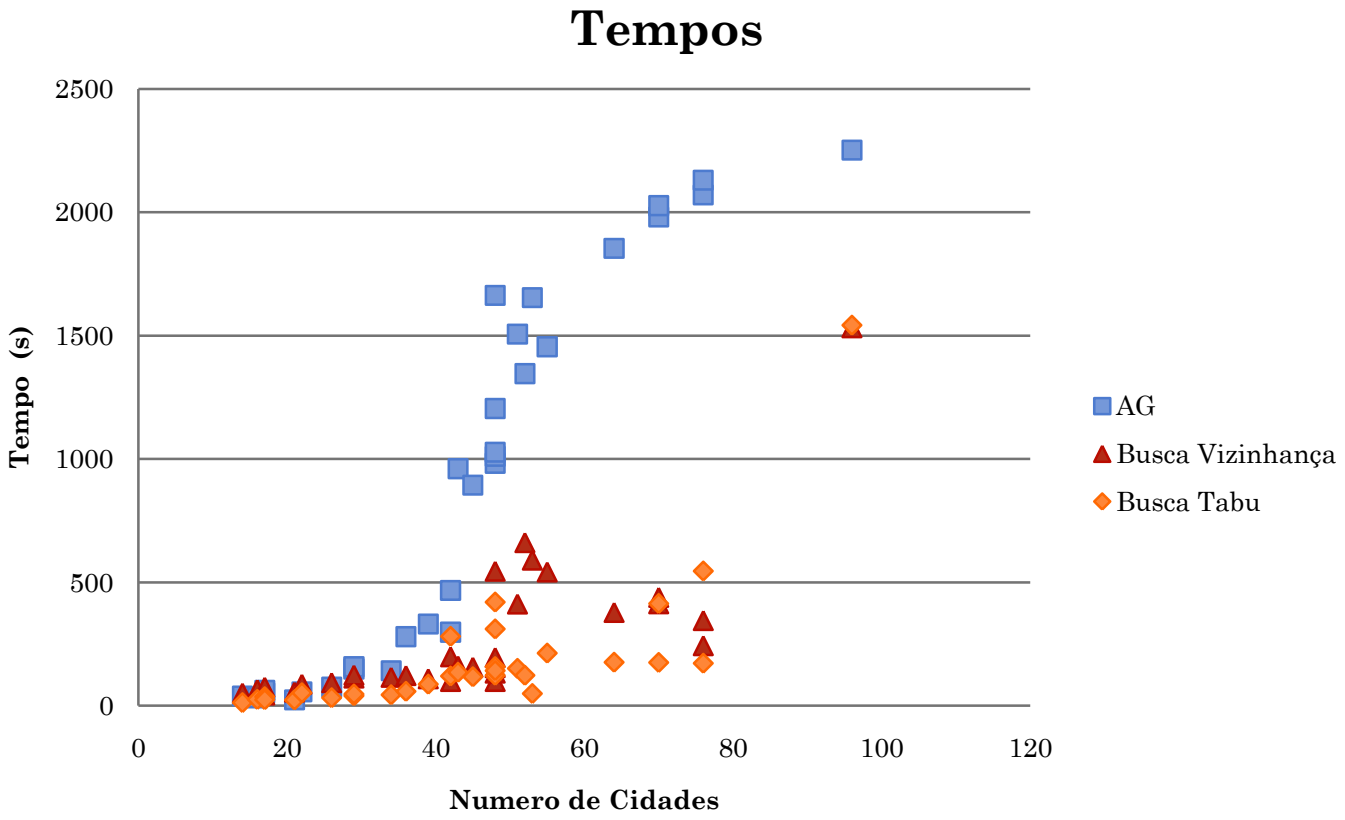


Resultados Computacionais

Instância	Numero de cidades	Valor ótimo	Algoritmo Genético		Busca por Vizinhança		Busca Tabu	
			Valor	Tempo (s)	Valor	Tempo (s)	Valor	Tempo (s)
burma14	14	3323	**3423**	39	3632	51	**3423**	12
ulysses16	16	6859	**6859**	32	**6859**	65	**6859**	25
br17	17	39	**39**	52	49	43	59	36
gr17	17	2085	2423	62	2097	75	2092	24
gr21	21	2707	**2707**	24	2707	55	2821	23
ulysses22	22	7013	7015	56	7115	87	7132	52
FRI26	26	937	974	76	1272	93	**937**	32
bayg29	29	1610	1789	149	1630	112	1612	40
bays29	29	2020	2457	158	2272	124	**2020**	47
ftv33	34	1286	1404	142	2681	115	1286	44
ftv35	36	1473	1565	280	1480	122	1490	58
ftv38	39	1530	1596	331	1645	109	1610	87
dantzig42	42	699	720	298	762	198	715	281
swiss42	42	1273	2393	467	2298	98	1379	120
p43	43	5620	5641	961	5758	161	5669	135
ftv44	45	1613	2503	894	1630	156	1736	117
ftv47	48	1776	1912	983	1881	98	1829	158
att48	48	10628	12632	1011	10646	154	12691	420
ry48p	48	14422	17913	1663	15520	195	14488	311
hk48	48	11461	12700	1205	11478	132	**11461**	123
gr48	48	5046	15001	1028	7133	545	5146	142
eil51	51	426	958	1506	1114	412	453	151
berlin52	52	7542	8742	1346	19564	661	9799	123
ft53	53	6905	8928	1654	7132	590	10993	49
ftv55	55	1608	1650	1455	2321	542	1608	213
ftv64	64	1839	2944	1854	1852	378	1890	176
ft70	70	38673	53636	1981	39677	413	42929	175
st70	70	675	**675**	2027	699	439	723	414
eil76	76	538	776	2071	593	243	589	546
pr76	76	108159	213355	2130	108168	345	17883	172
gr96	96	55209	69123	2252	65126	1532	55324	1543


Resultado busca tabu
250000
200000
150000
100000
50000
0
Título do Eixo
0
20
40
60
80
100
120
Título do Eixo
Resultado ótimo
AG
Série4
Série1

Tempos Computacionais



Grandes Dificuldades

- Evitar repetições de soluções nas soluções finais , pois caso contrário a solução final tende à uma população formada por rotas iguais.

- No caso específico dos Algoritmos Genéticos, realizar um algoritmo inteligente de forma que as operações de cruzamento e mutação sempre originasse rotas possíveis para a população.

- Problemas com o consumo de memória. Algumas meta-heurísticas usam uma estrutura estática em seus processos, o que pode gerar um tempo maior de processamento, visto que ela utiliza um espaço pré determinado mesmo não sendo utilizado todo espaço.

- Tempo excessivamente longo na execução das meta-heurísticas.

- Porém, problemas como valores obtidos com as meta-heuristicas estavam longe do ótimo, devido ao fato de escolhas prévias como, “qual o numero máximo de iterações deve ser utilizada para um número específico de cidades”. Ainda continuavam impactando nos resultados finais.

Como esses parâmetros podem influenciar?

Instância	Numero de Cidades	Valor Ótimo	Algoritmo Genético		
			Tamanho da populacao	Numero de iterações	Resultado ótimo
gr17	17	2085	4	20	3950
gr17	17	2085	4	50	3282
gr17	17	2085	10	20	3215
gr17	17	2085	10	50	3215
gr17	17	2085	4	100	3282
gr17	17	2085	10	100	29773
gr17	17	2085	10	1000	2449
gr17	17	2085	10	2000	2085
bays29	29	2020	5	40	4696
bays29	29	2020	10	40	4579
bays29	29	2020	10	100	4054
bays29	29	2020	20	100	4097
bays29	29	2020	30	2000	2491
bays29	29	2020	40	3500	2390
bays29	29	2020	80	6500	2025
Ft53	53	6905	5	50	14696
Ft53	53	6905	50	5000	13212
Ft53	53	6905	250	5000	15472
Ft53	53	6905	150	7000	9735
Ft53	53	6905	250	200000	7120
krA100	100	21282	50	100	149650
krA100	100	21282	100	1000	124625
krA100	100	21282	200	10000	104137
krA100	100	21282	500	100000	67772
krA100	100	21282	1000	150000	22520
Ftv33	33	1286	4	40	3102
Ftv33	33	1286	10	400	2445
Ftv33	33	1286	20	4000	1832
Ftv33	33	1286	120	90000	1699
FRI26	26	937	4	20	1286
FRI26	26	937	10	50	1872
FRI26	26	937	10	200	1325
FRI26	26	937	15	4000	937

Instância	Numero de Cidades	Valor Ótimo	Busca Tabu	
			Numero de iterações	Resultado ótimo
gr17	17	2085	100	4982
gr17	17	2085	500	4870
gr17	17	2085	1000	3560
gr17	17	2085	5000	2870
gr17	17	2085	5500	2270
bays29	29	2020	40	4769
bays29	29	2020	80	4579
bays29	29	2020	100	4054
bays29	29	2020	1000	3597
bays29	29	2020	2000	2950
bays29	29	2020	3500	2250
bays29	29	2020	6500	2025
Eil50	50	6905	50	10696
Eil50	50	6905	5000	8953
Eil50	50	6905	5000	7921
Eil50	50	6905	7000	7120
krA100	100	21282	100	129556
krA100	100	21282	3000	80329
krA100	100	21282	4000	53291
krA100	100	21282	50000	32191
krA100	100	21282	180000	22191
FRI26	26	937	20	1468
FRI26	26	937	50	1271
FRI26	26	937	100	1024
FRI26	26	937	200	937

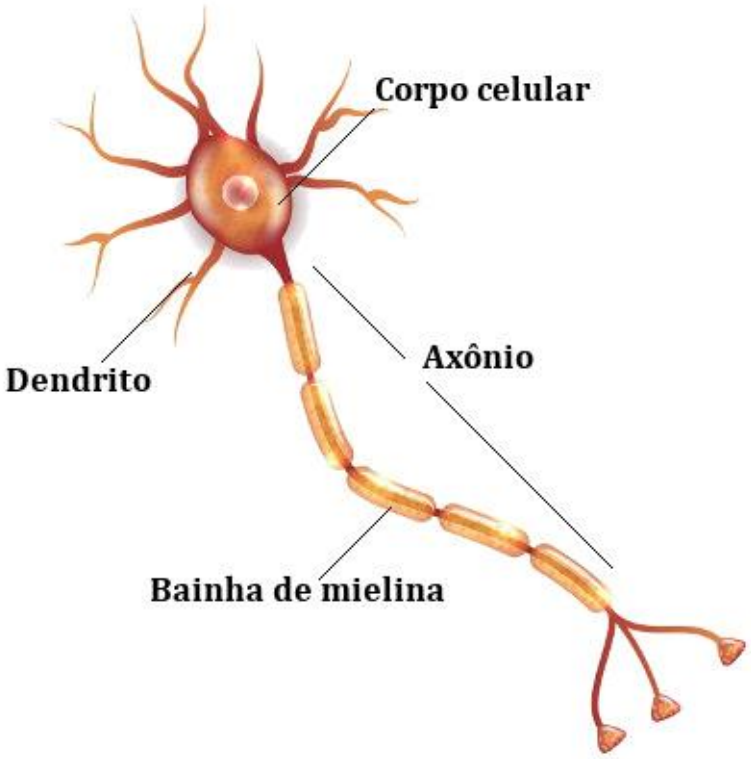
Instância	Numero de Cidades	Valor Ótimo	Busca Por Vizinhança	
			Numero de iterações	Resultado ótimo
gr17	17	2085	100	8790
gr17	17	2085	500	8270
gr17	17	2085	10000	3560
gr17	17	2085	50000	2870
gr17	17	2085	155000	2270
bays29	29	2020	1000	7953
bays29	29	2020	100000	6597
bays29	29	2020	650000	2027
Eil50	50	6905	5000	12696
Eil50	50	6905	50000	8563
Eil50	50	6905	800000	7220
krA100	100	21282	100	177557
krA100	100	21282	30000	80329
krA100	100	21282	40000	63231
krA100	100	21282	150000	35496
krA100	100	21282	2800000	22191
FRI26	26	937	200	3468
FRI26	26	937	5000	2218
FRI26	26	937	10000	1134
FRI26	26	937	200000	937

Rede Neural

- As Redes Neurais Artificiais (RNA) inspiram-se em modelos biológicos, são técnicas computacionais que apresentam um modelo matemático inspirado na estrutura neural que é capaz de adquirir conhecimento de experiências.

- Os neurônios se comunicam através de sinapses. A sinapse é uma região onde dois neurônios entram em contato e são transmitidos os impulsos nervosos entre eles

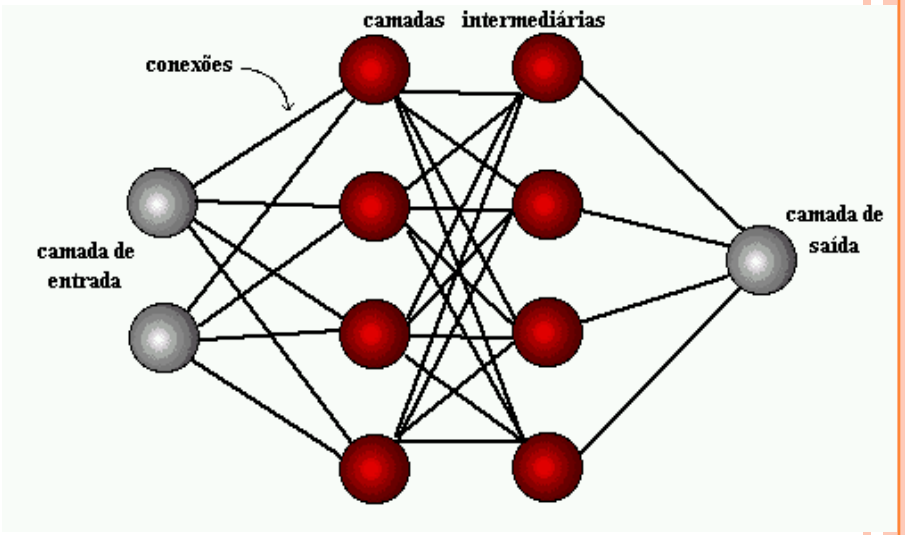
- Os impulsos recebidos por um neurônio, em um determinado momento, são processados até que se atinjam um limiar de excitação ou ação. Assim, esse neurônio produz uma substância neurotransmissora que flui do corpo celular para o axônio, sendo enviado finalmente a outro neurônio ligados a ele pelo axônio.



Funcionamento

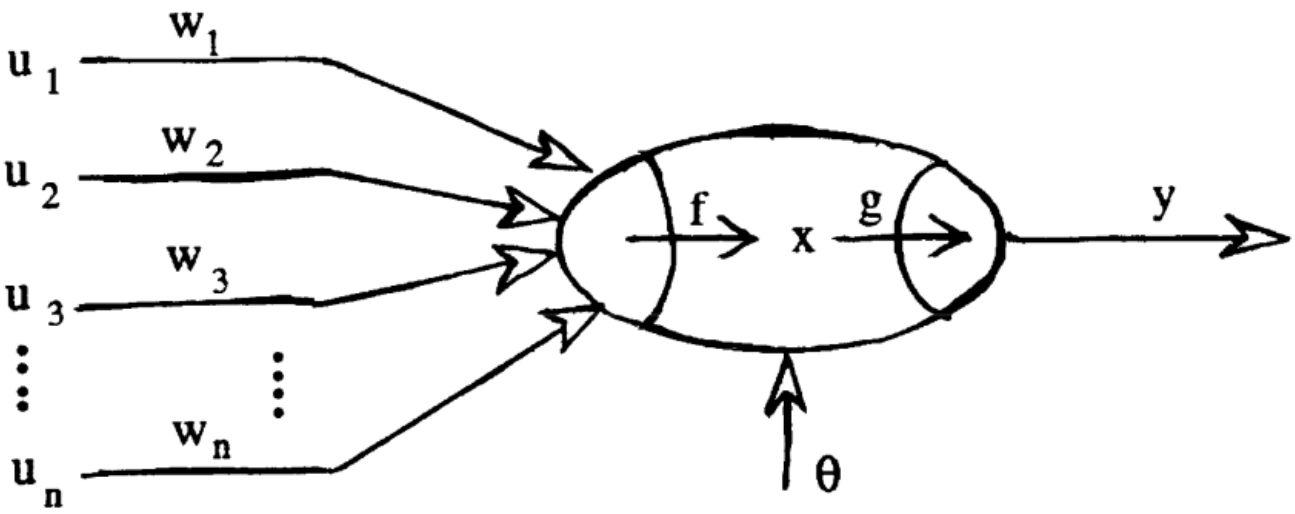
- A cada entrada é multiplicado por um número, ou peso, que indica a sua influência na saída da unidade. Então, esses valores são somados na função pode ser representado na equação

$$u = \sum_{i=0}^{n} W_i X_i$$



- Cujo X_i representa sinais de entrada.
- W_i representa sinapses.
- u representa sinal de saída.

- A função f(x) normalmente é um somatório.
- A função g(x) é chamada de função para transferência de sinais entre neurônios, também chamada de função de limiar.



Processo de aprendizagem

- Supervisionado: neste tipo de aprendizado são apresentados à rede os padrões de entrada e saída. O treinamento supervisionado consiste em verificar se a saída obtida da rede foi a mesma da desejada ou próxima o suficiente que possa ser aceita por um erro mínimo. Se o erro for maior que o aceitável, um ajuste de pesos é feito na rede.

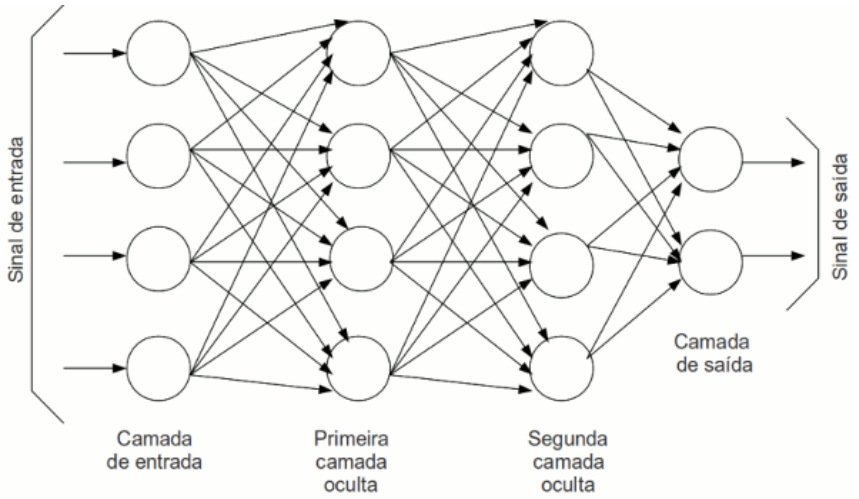
- Não-supervisionado: não existe a indicação da resposta desejada para o padrão da entrada, e sim a indicação de quão semelhante. A partir destas propriedades é que o aprendizado é constituído.

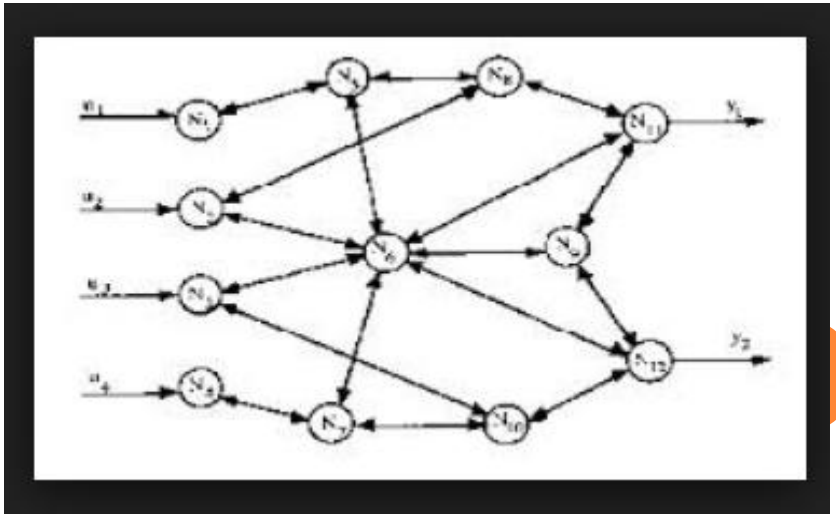
- Híbrido: neste tipo ocorre uma "mistura" dos tipos supervisionado e não-supervisionado, sendo que uma camada trabalha com um tipo de aprendizado enquanto outra realiza o outro aprendizado.

Conexões

- Redes diretas (*feedforward*): Esses tipos de conexão geralmente não possuem nenhuma forma de ciclo. Geralmente apresenta as camadas de entrada, camadas internas (*hidden*) e as camadas de saída

- Redes com *feedback*: São redes que possuem ciclos em suas conexões





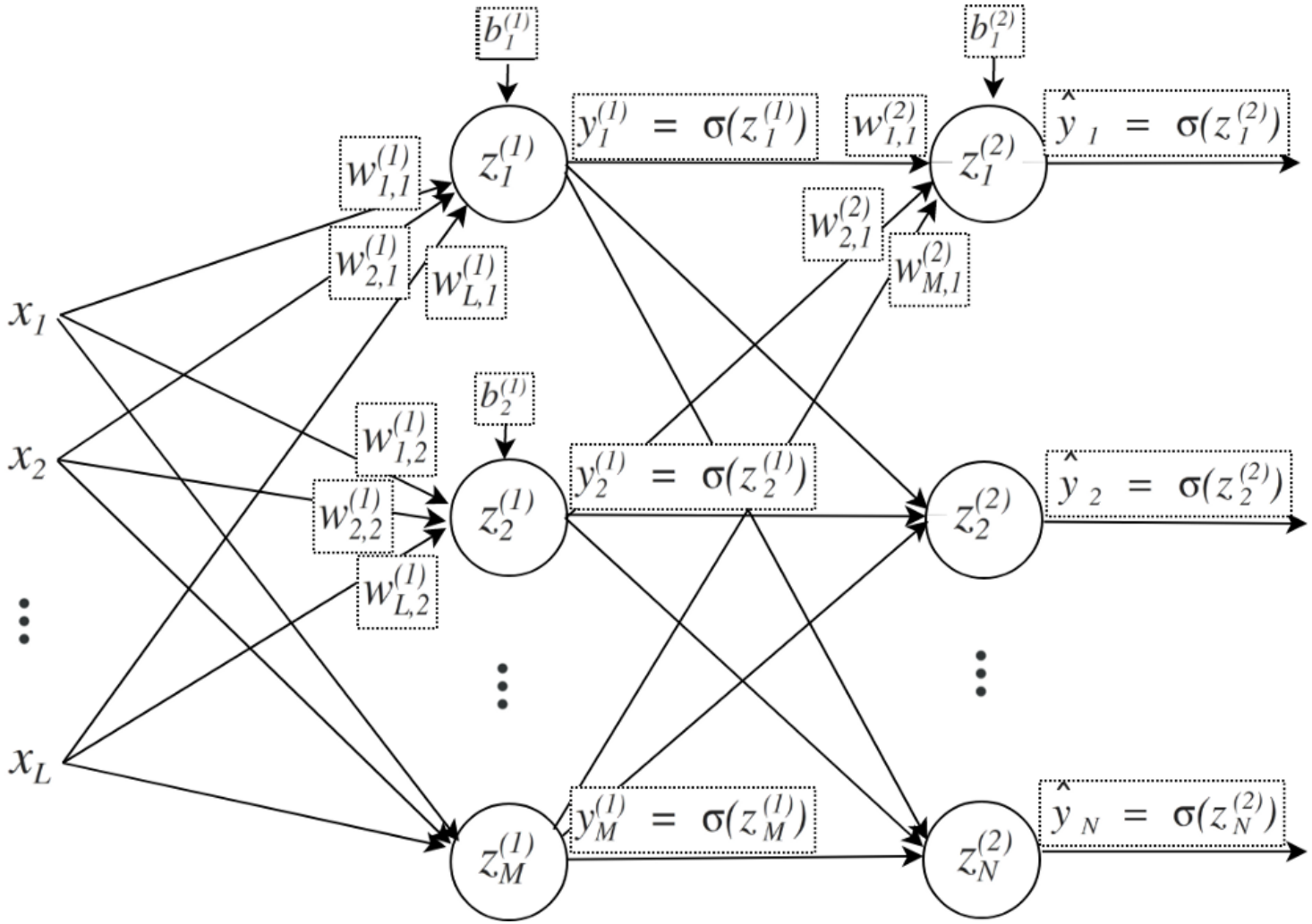
BackPropagation

- A rede *backpropagation,* ou retropopagação, é uma rede neural que possui conexões do tipo *feedforward* ou direta e possui uma arquitetura multicamada com treinamento supervisionado.

- O backpropagation possui 2 passos:processo direto, as quais uma entrada tem seu efeito aplicado em toda rede, e no ajuste de pesos no processo reverso, o qual o erro calculado na saída da rede é propagado no sentido reverso de acordo com a equação

$$w = w - n\frac{\partial E}{\partial w}$$

Exemplo de Rede



PYBRAIN

- implementação de uma rede neural *Perceptron* com *Backpropagation* com o auxilio da biblioteca *pybrain*

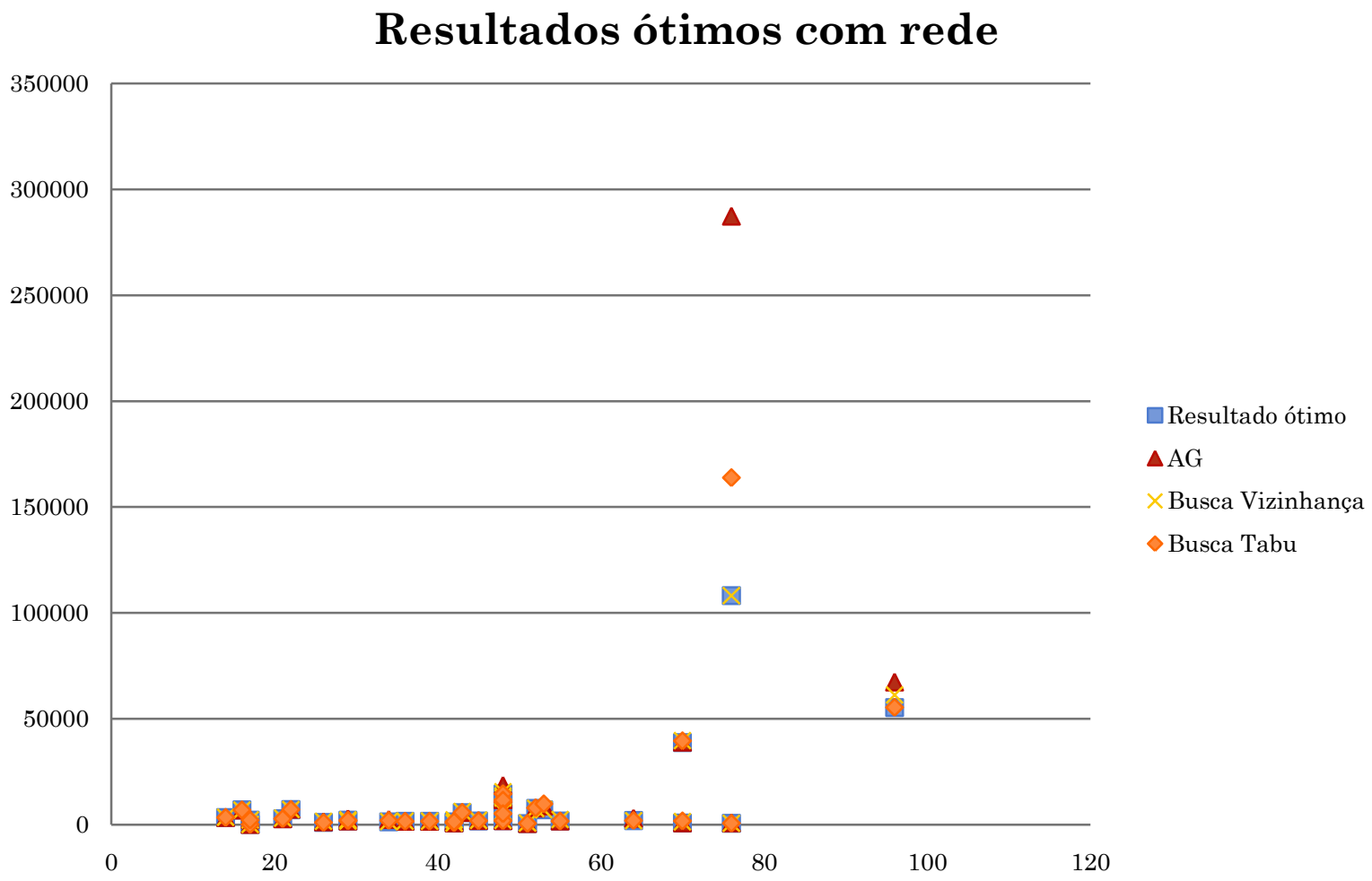
```
from pybrain.tools.shortcuts import buildNetwork
#from pybrain.datasets import SupervisedDataSet
from pybrain.datasets import SupervisedDataSet
#from pybrain.tools.shortcuts import buildNetwork
from pybrain.supervised.trainers.backprop import BackpropTrainer
class rede():
    def redea(self):
        rede = buildNetwork(2,4,1,bias=True)

        base = SupervisedDataSet(2, 1)
        .....
        base.addSample((redevalores[i],1), (rederesultado[i]))

        treinamento = BackpropTrainer(rede, base)
        for i in range(1000):
            print(treinamento.train())
        print('entrada')
        print(base['input'])
        print('\nsaida\n')
        print(base['target'])
        while True:
            numerodecidade = float(input('\nNumero de cidades: '))
            z = rede.activate((numerodecidade,1))
```

Sistema Híbrido

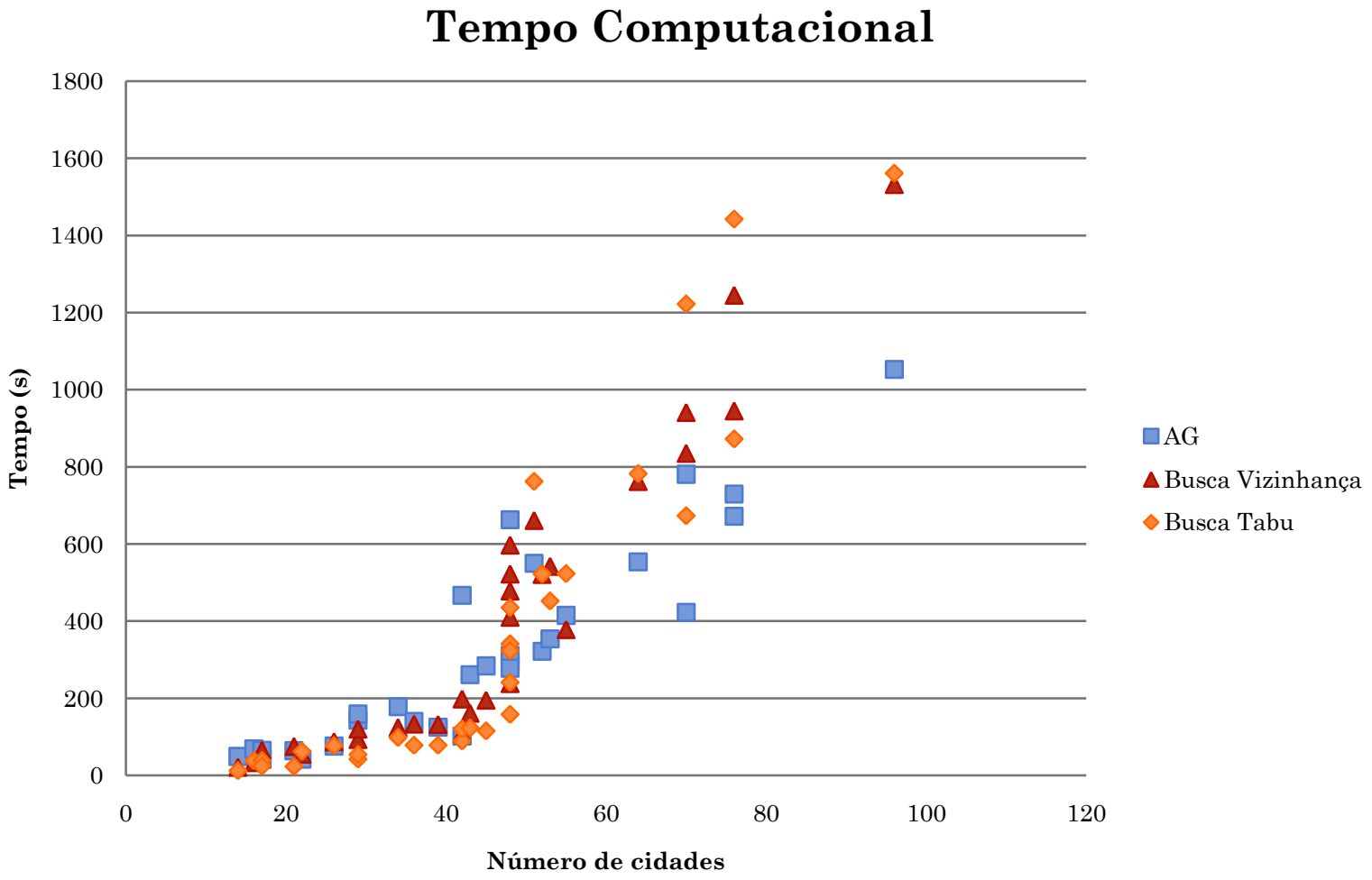
- Os sistemas híbridos podem ser caracterizados como uma combinação de sub-sistemas, em que cada um possui uma identidade individual e não apresentem características não-coincidentes entre si
- Essa combinação deve originar um método que possua um potencial de solucionar o problema maior do que a soma dos potencias de seus sub-sistemas isolados.
  - Não existe um único método que seja ótimo para qualquer problema.
  - Além disso, deixar um único método dedicado à solução pode ser um processo mais custoso do que a obtenção de uma solução pela combinação de múltiplos métodos

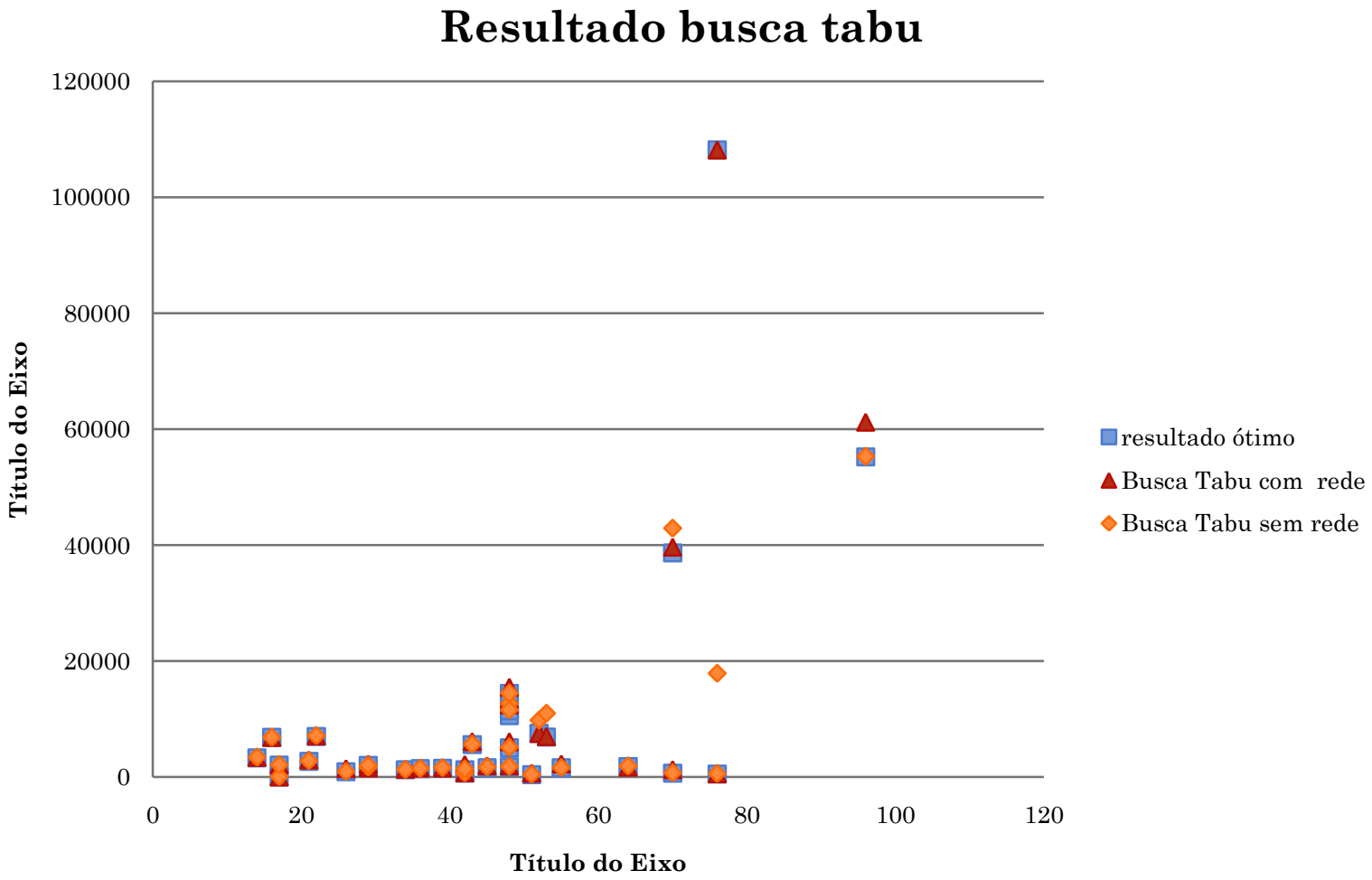
Resultados Computacionais com Rede

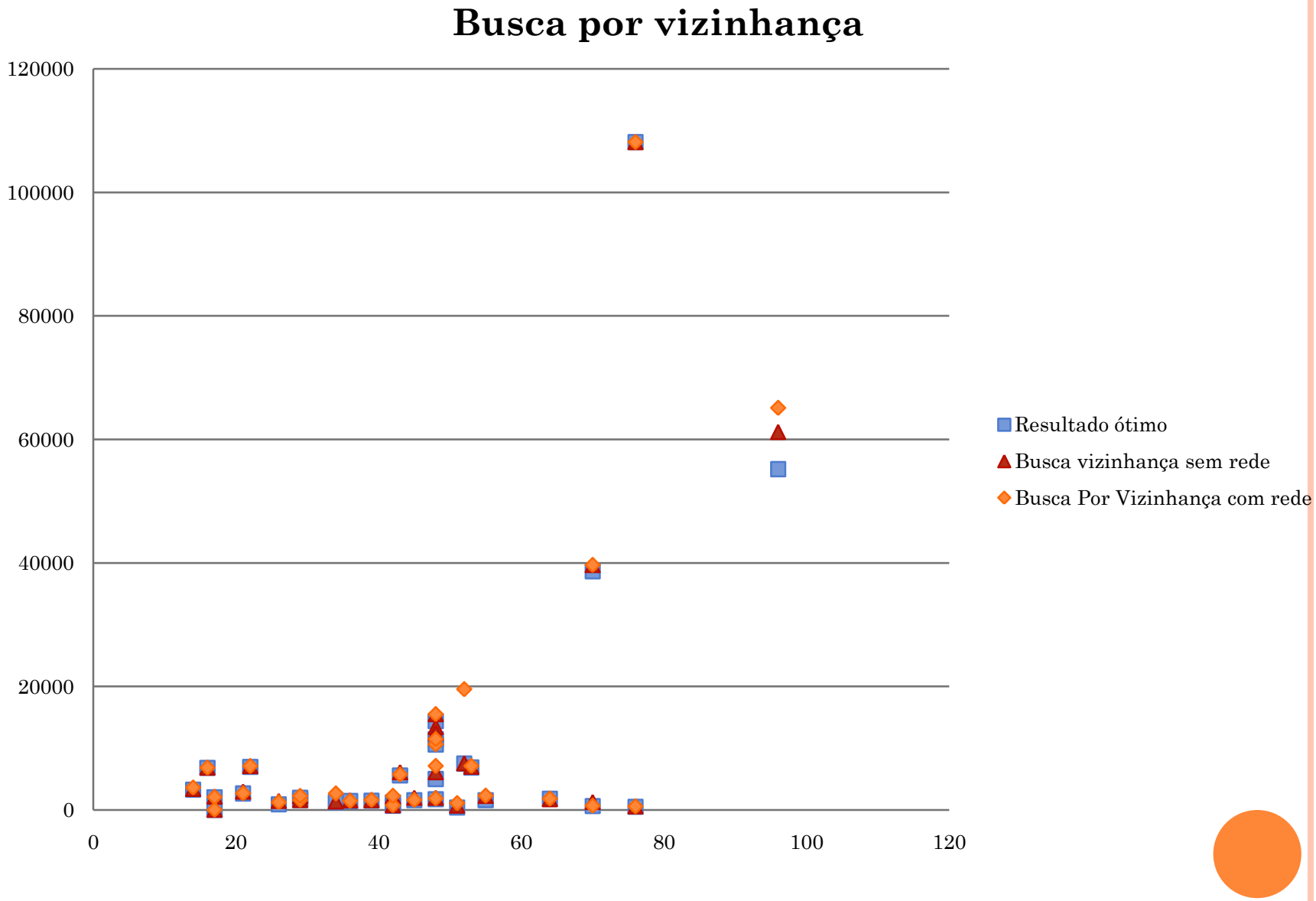
Instância	Numero de cidades	Valor ótimo	Algoritmo Genético		Busca por Vizinhança		Busca Tabu	
			Valor	Tempo (s)	Valor	Tempo (s)	Valor	Tempo (s)
burma14	14	3323	**3323**	49	3432	21	**3323**	12
ulysses16	16	6859	**6859**	69	6859	33	**6859**	38
br17	17	39	**39**	44	45	42	**39**	38
gr17	17	2085	2422	65	2175	65	**2085**	24
gr21	21	2707	**2707**	64	2982	75	**2707**	23
ulysses22	22	7013	7240	42	7125	55	**7013**	62
FRI26	26	937	**937**	76	1472	87	**937**	77
bayg29	29	1610	1657	143	1620	93	1722	42
bays29	29	2020	**2020**	159	2241	120	**2020**	54
ftv33	34	1286	2409	178	1381	124	1885	98
ftv35	36	1473	1532	140	1580	133	1480	78
ftv38	39	1530	1539	125	1596	132	**1530**	78
dantzig42	42	699	765	102	782	109	**699**	89
swiss42	42	1273	1492	467	2198	198	**1273**	120
p43	43	5620	5691	261	6125	161	5719	123
ftv44	45	1613	1892	284	1982	195	1783	115
ftv47	48	1776	1821	287	1982	238	1794	158
att48	48	10628	12732	311	13422	410	10722	435
ry48p	48	14422	14475	663	15520	597	14476	341
hk48	48	11461	11732	304	12478	522	11482	323
gr48	48	5046	14401	278	6120	478	5098	241
eil51	51	426	**426**	550	714	661	**426**	762
berlin52	52	7542	7782	322	7564	521	**7542**	522
ft53	53	6905	8767	354	7016	542	9727	452
ftv55	55	1608	1650	415	2286	378	**1608**	523
ftv64	64	1839	2944	554	1752	762	**1839**	782
ft70	70	38673	38909	781	39677	941	39529	673
st70	70	675	784	423	1299	835	1675	1222
eil76	76	538	776	672	593	945	**538**	1442
pr76	76	108159	287318	730	108168	1245	163883	872
gr96	96	55209	67289	1053	61228	1532	55324	1561

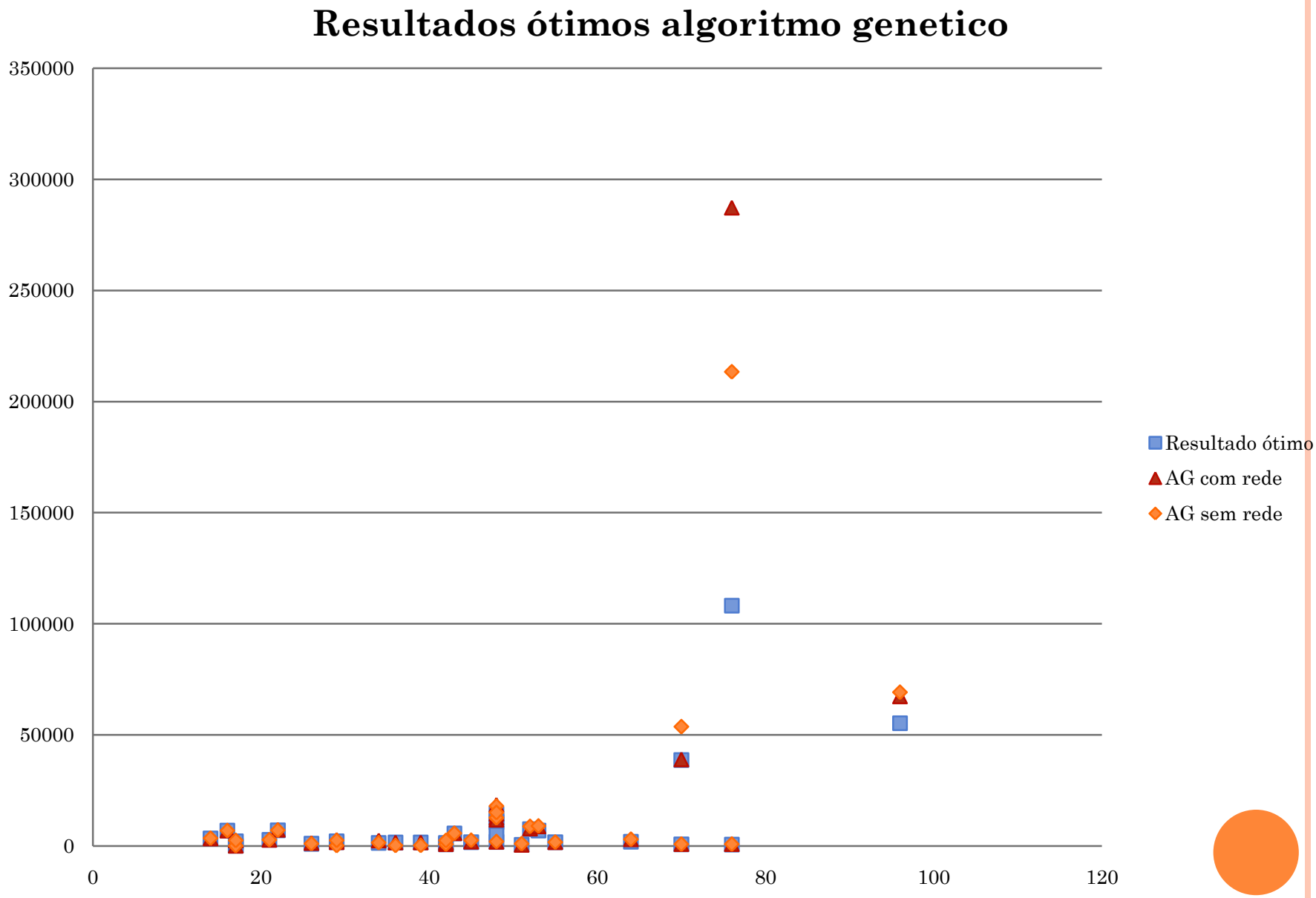

Resultados ótimos com rede
350000
300000
250000
200000
150000
100000
50000
0
0
20
40
60
80
100
120
Resultado ótimo
AG
Busca Vizinhança
Busca Tabu

Tempo Computacional com Rede




Resultado busca tabu
120000
100000
80000
60000
40000
20000
0
Título do Eixo
0
20
40
60
80
100
120
Título do Eixo
resultado ótimo
Busca Tabu com rede
Busca Tabu sem rede


Busca por vizinhança
120000
100000
80000
60000
40000
20000
0
0
20
40
60
80
100
120
Resultado ótimo
Busca vizinhança sem rede
Busca Por Vizinhança com rede


Resultados ótimos algoritmo genetico
350000
300000
250000
200000
150000
100000
50000
0
0
20
40
60
80
100
120
Resultado ótimo
AG com rede
AG sem rede

Considerações sobre Algoritmo Genético

- Algoritmos Genéticos possui uma grande dificuldade em determinar o ótimo global exato, quando o numero de variáveis (cidades envolvidas) aumenta. Isso ocorre, principalmente, pelo método utilizado, como *crossover* e mutação, que requerem um grande numero de iteração para que possa gerar soluções aptas e viáveis a solução final.

- Além disso, o mal dimensionamento do número de população bem como o número de iterações também pode impactar no resultado final

- Apesar disso, observa-se que este método possui vantagens, entre as quais pode-se citar:
  - É um método robusto e aplicável para uma quantidade muito variada de problemas.
  - São de fácil implementação e por isso proporcionam uma maior flexibilidade para tratamento de problemas.
  - Consegue convergir rapidamente para um resultado melhor que o inicial

Considerações sobre Busca por Vizinhança

- O método apresentou um bom desempenho em relação a proximidade do ótimo global. Porém, esse método não deve ser utilizado para testes que possuem muitas instancias, visto que é um algoritmo que lida com trocas sistemática (espécie de "força bruta") e à medida que a vizinhança estudada aumentar ela pode cair em trocas já realizadas anteriormente, que foram descartadas, ou realizar troca já vigentes na solução fazendo assim mudadas desnecessários

- Com o uso da Rede Neural para a Busca por Vizinhança, também houve uma melhora dos resultados finais, porem devido o aumento do número de iterações o tempo de processamento aumentou, tornado o uso dessa meta-heuristica inviável

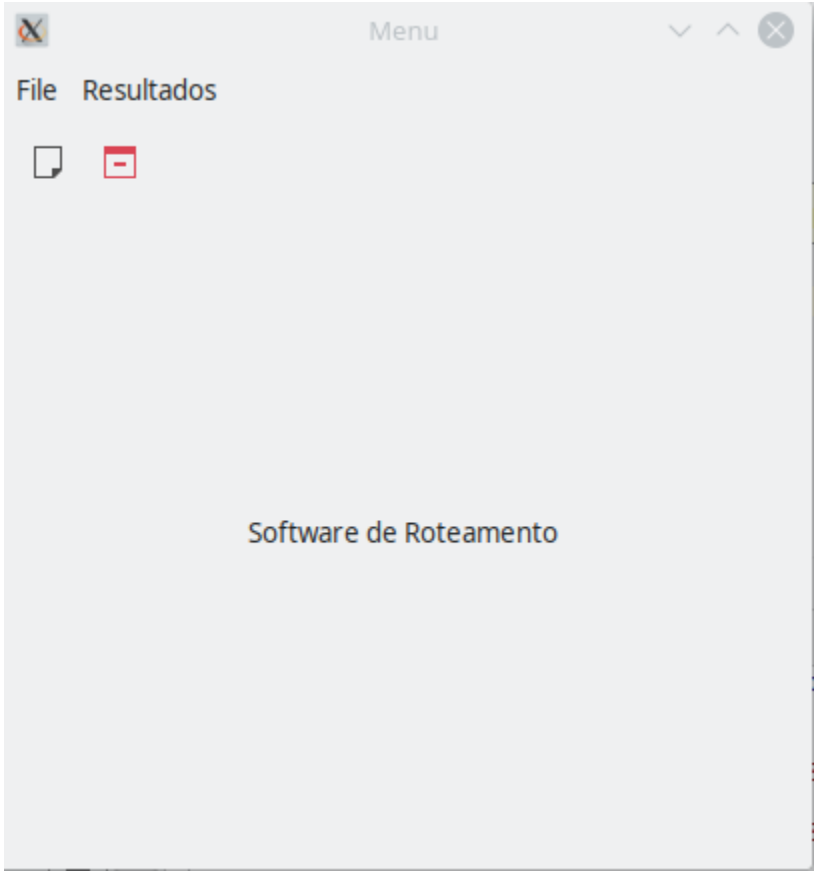
Considerações Busca Tabu

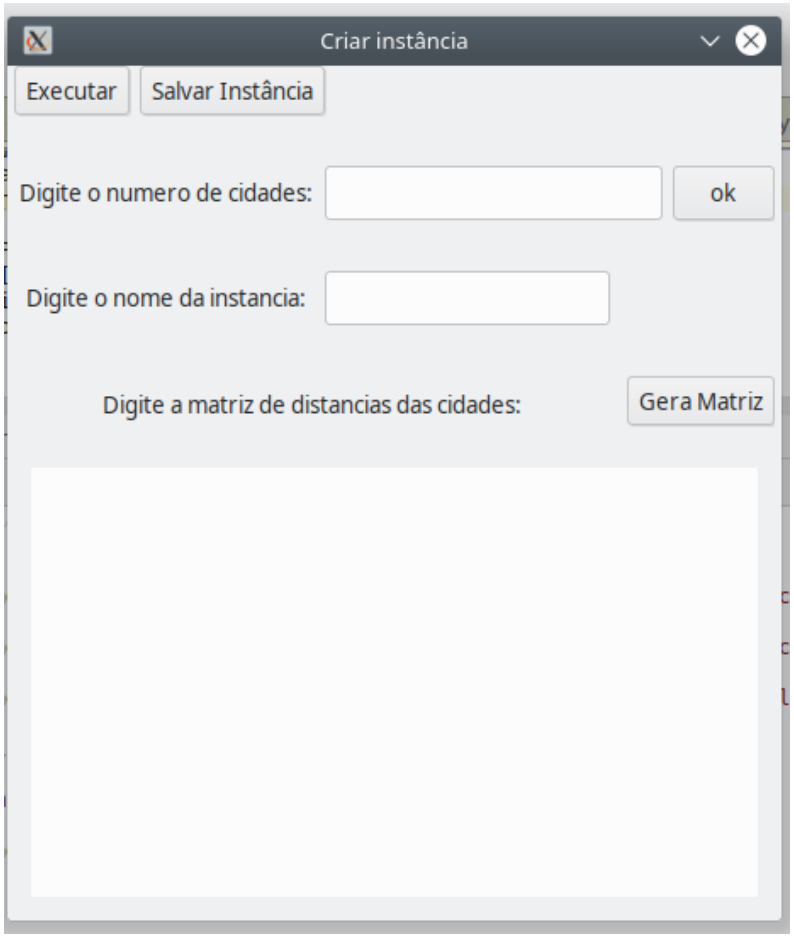
- Esta meta-heurística foi a que apresentou melhores resultados em comparação às outras apresentadas neste estudo. Devido ao fato dela ser uma variação do busca por vizinhança, facilita em não cair em uma busca cíclica de um ótimo global (devido à criação de suas listas adaptativas).

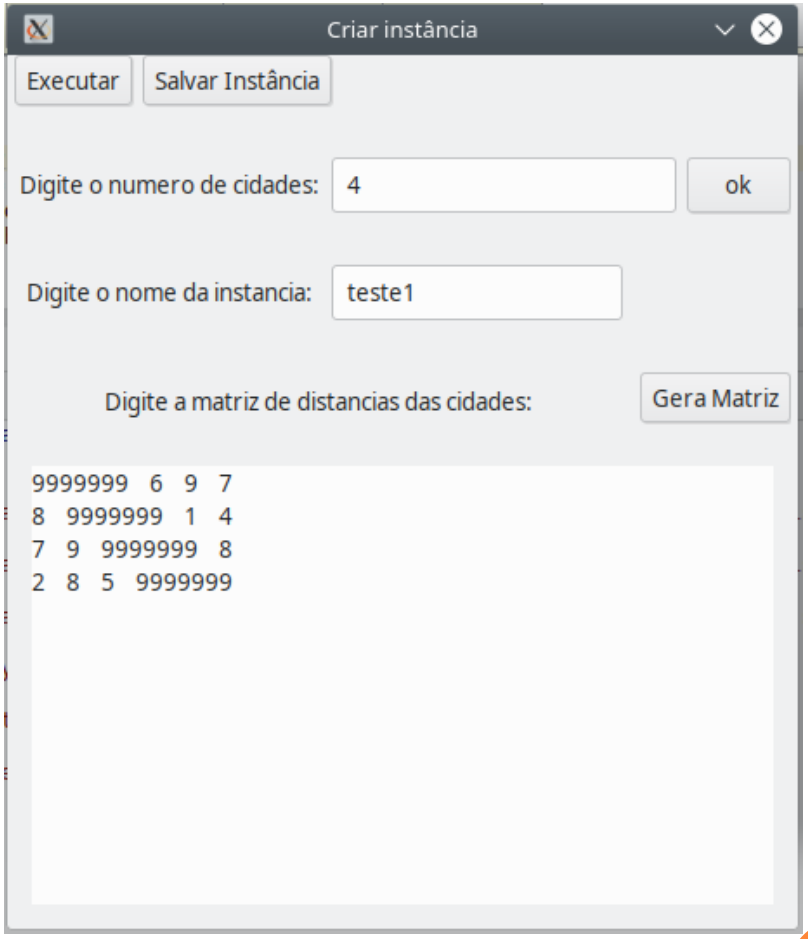
- Com um numero muito grande de cidades também aumenta as chances de espaços salvos na lista tabu serem descartados depois fazendo assim um uso desnecessário de memória.

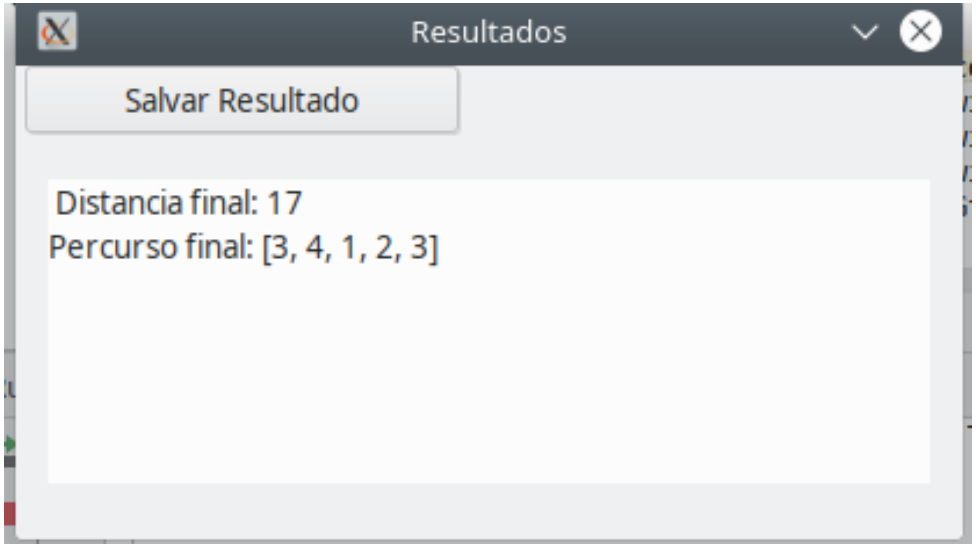
- O mal dimensionamento do numero de iterações pode acarretar em um resultado muito longe do ideal.

Software




Criar instância
Executar
Salvar Instância
Digite o numero de cidades:
ok
Digite o nome da instancia:
Digite a matriz de distancias das cidades:
Gera Matriz


Criar instância
Executar
Salvar Instância
Digite o numero de cidades:
4
ok
Digite o nome da instancia:
teste1
Digite a matriz de distancias das cidades:
Gera Matriz
9999999 6 9 7
8 9999999 1 4
7 9 9999999 8
2 8 5 9999999


Resultados
Salvar Resultado
Distancia final: 17
Percurso final: [3, 4, 1, 2, 3]

Considerações Finais

- Observou-se que estes algoritmos são suficientemente complexos e muito eficiente para busca de soluções ótimas, ou quase-ótimas, pois utilizam mecanismos de busca adaptativo poderosos e robustos, devido ao fato que pode-se encontrar diversos problemas para encontrar o ótimo global.

- Observa-se também, pelos resultados dos testes, que a Busca Tabu foi a que mais se aproximou dos resultados ótimos em um tempo de processamento razoavelmente "aceitável", sendo a mais indicada dentre as que foram desenvolvidas para a resolução do Problema do Caixeiro Viajante.

- Conclui-se ainda que, com uso da rede neural todos os métodos melhoraram seus resultados ótimos, pois através dela foi possível dimensionara melhor algumas variáveis que impactavam no bom funcionamento do algoritmo.

- Apesar disso, o tempo dos métodos aumentaram pelo fato do treinamento da rede e por realizar mais interações de acordo com os novos valores atribuído pela rede. Além disso, deixar um único método dedicado à solução pode ser um processo mais custoso do que a obtenção de uma solução pela combinação de múltiplos métodos

Agradecimentos