Instituto Politécnico de Coimbra

Instituto Superior de Engenharia de Coimbra

Departamento de Engenharia Informática e de Sistemas

Previsão da Estrutura de Proteínas com Modelos HP

Cátia Regina Craveiro Pires

Mestrado em Informática e Sistemas

Coimbra, dezembro 2015

Instituto Politécnico de Coimbra

Instituto Superior de Engenharia de Coimbra

Departamento de Engenharia Informática e de Sistemas

Unidade Curricular Estágio/Projecto Industrial

Previsão da Estrutura de Proteínas com Modelos HP

Cátia Regina Craveiro Pires

Orientador: Prof. Doutor Francisco José Baptista Pereira

Mestrado em Informática e Sistemas

Coimbra, dezembro 2015

Agradecimentos

Começo por agradecer ao meu orientador do DEIS, Professor Francisco Pereira por todo o conhecimento que me transmitiu, pela paciência e ajuda prestada durante todo o projecto.

Agradeço aos meus colegas de curso assim como à minha família e amigos, pela ajuda e motivação dada durante o decorrer do projecto.

Um grande obrigado a todos os que me ajudaram a tornar este projecto possível.

Resumo

O presente documento foi realizado no âmbito da unidade curricular Estágio/Projecto Industrial do segundo ano de Mestrado em Informática e Sistemas – ramo Desenvolvimento de Software do Instituto Superior de Engenharia de Coimbra.

A previsão da estrutura de proteínas é um problema importante e de difícil resolução. Este problema consiste na obtenção da estrutura tridimensional de uma proteína a partir da sua sequência de aminoácidos. A sua resolução pode ser a chave para o conhecimento da origem de certas doenças, e um grande passo para a sua cura.

Uma das possíveis abordagens para resolver este problema é recorrer a simulações computacionais que efectuem a previsão da estrutura tridimensional. No contexto deste trabalho, estas simulações computacionais são feitas recorrendo a modelos HP que simplificam a cadeia de aminoácidos e o processo de criação da estrutura tridimensional. A obtenção da estrutura é feita recorrendo a algoritmos de optimização baseados em colónias de formigas.

No âmbito deste projecto foram desenvolvidos e testados algoritmos de optimização baseados em colónias de formigas. As variantes implementadas foram: o Ant System, Elitist Ant System, Rank-based Ant System e o Max-Min Ant System. Estes algoritmos foram aplicados aos modelos HP para a previsão de estrutura de proteínas. Com base neste projecto foi ainda desenvolvida uma aplicação web, que pode ser utilizada para fazer a previsão da estrutura de proteínas.

Abstract

This document was prepared as part of the course Estágio/Projecto, from the second year of the Master Degree in Computer and Systems - Software Development branch of Instituto Superior de Engenharia de Coimbra.

Protein structure prediction is an important and difficult problem to solve. This problem consists in obtaining the three dimensional structure of a protein from its amino acid sequence. Solving this problem can be the key to a better understanding of the origin of certain diseases, and a big step towards its cure.

One possible approach to solve this problem is to use computer simulations that predict the three-dimensional structure. In the context of this work, these computer simulations are made using HP models that simplify the chain of amino acids and the process of creating the three-dimensional structure. The structure is obtained using optimization algorithms based on ant colonies.

The objectives of this project were to develop and test optimization algorithms based on ant colonies. Serveral variants were implemented: the Ant System, Elitist Ant System, Rank-based Ant System and the Max-Min Ant System. These algorithms were applied to HP models for predicting protein structure. Based on this project, we developed an application to make the prediction of the protein structure.

Índice

Agradecimentos ..... *i*

Resumo ..... *ii*

Abstract ..... *iii*

Índice ..... *iv*

Índice de Figuras ..... *vi*

Índice de Tabelas ..... *viii*

Definições e Abreviaturas ..... *ix*

1 Introdução *1*

1.1 Contexto do Problema 1

1.2 Contexto e Objectivos do Projecto 1

1.3 Estratégia de Desenvolvimento 2

1.4 Estrutura do Relatório 3

2 Optimização baseada em colónias de formigas *4*

2.1 Introdução 4

2.2 Modelos computacionais baseados em colónias de formigas 5

2.3 Variantes de algoritmos baseados em colónias de formigas 8

2.3.1 Ant System 8

2.3.2 Elitist Ant System 10

2.3.3 Rank-based Ant System 10

2.3.4 Max-Min Ant System 11

2.4 Parâmetros 12

2.5 Áreas de aplicação 12

3 Modelos HP *14*

3.1 Introdução 14

3.1 Modelos HP 16

3.2 Outras abordagens para a resolução de modelos HP 18

4 Aplicação de ACO a modelos HP *20*

4.1 Como aplicar um ACO a um problema HP 20
4.2 Testes 21
4.2.1 Resultados dos Testes aplicados à Sequência 1 22
4.2.2 Resultados dos Testes aplicados à Sequência 2 24
4.2.3 Resultados dos Testes aplicados à Sequência 3 27
4.2.4 Resultados dos Testes aplicados à Sequência 4 31
4.2.5 Testes complementares 34
4.3 Análise 38
5 Aplicação 42
5.1 Descrição Geral 42
5.2 Funcionalidades 42
5.3 Interfaces da aplicação 45
5.3.1 Resolver Problema 45
5.2.1 Páginas adicionais 53
6 Conclusão 57
6.1 Resultados do Projecto 57
6.2 Trabalho Futuro 58
6.3 Apreciação Crítica do Estágio 58
Referências 60
Anexos 63

Índice de Figuras

Figura 1 - Comportamento natural das formigas 4
Figura 2 – Exemplo da representação de um grafo 6
Figura 3 – Escolha aleatória da cidade 6
Figura 4 – Cálculo do potencial de cada ligação 6
Figura 5 – Uma possível solução para o TSP 7
Figura 6 – Ligações possíveis a partir do vértice i 9
Figura 7- Esquema da Transcrição e Tradução do DNA 14
Figura 8- Nível estrutural das proteínas 15
Figura 9 – Passos de construção de uma solução 17
Figura 10 – Exemplo de soluções de uma determinada sequência 18
Figura 11 – Solução inválida 20
Figura 12 - Heurística 21
Figura 13 - Resultados obtidos da aplicação do AS sem heurística com 100 iterações 25
Figura 14 - Resultados obtidos da aplicação do AS com heurística com 100 iterações 26
Figura 15 - Resultados obtidos da aplicação do RAS sem heurística com 50 iterações 26
Figura 16- Resultados obtidos da aplicação do RAS com heurística com 50 iterações 27
Figura 17 – Percentagem de inválidos na última iteração de experiências com e sem heurística 29
Figura 18 - Resultados obtidos da aplicação do EAS sem heurística com 100 iterações 30
Figura 19 - Resultados obtidos da aplicação do EAS com heurística com 100 iterações 30
Figura 20 - Percentagem de inválidos na última iteração de experiências com e sem heurística 32
Figura 21 - Resultados obtidos da percentagem de inválidos em relação ao número de iterações, sem heurística 33
Figura 22 - Resultados obtidos da percentagem de inválidos em relação ao número de iterações, com heurística 33
Figura 23 – Cruzamento dos dados obtidos relativamente ao Ant System 39
Figura 24 - Cruzamento dos dados obtidos relativamente ao Elitist Ant System 39
Figura 25 - Cruzamento dos dados obtidos relativamente ao Rank-based Ant System 40
Figura 26- Cruzamento dos dados obtidos relativamente ao Max-Min Ant System 41
Figura 27 – Interface gráfica da página Problema.jsp 45
Figura 28 – Opções para inserir sequência 46
Figura 29 – Opção “Sequência aleatória” seleccionada 46
Figura 30 – Opção “Inserir Sequência” seleccionada 46
Figura 31 - Opção “Fazer upload de ficheiro” seleccionada 47
Figura 32 – Parâmetros do Algoritmo 47
Figura 33 – Opções do tipo de algoritmo 47
Figura 34 - Parâmetros 48
Figura 35 – Esquema de funcionamento da resolução de um problema 49

Figura 36 – Matriz da melhor solução 50
Figura 37 – Matriz de Feromonas..... 50
Figura 38 – Visualizar soluções..... 51
Figura 39 – Pdf gerado pela aplicação 52
Figura 40 – Gráfico gerado pela aplicação..... 53
Figura 41 – Interface gráfica da página Início.jsp..... 53
Figura 42 – Interface gráfica da página ACO.jsp 54
Figura 43 – Interface gráfica da página ModelosHP.jsp 54
Figura 44 – Interface gráfica da página AntSystem.jsp..... 55
Figura 45 – Interface gráfica da página Algoritmos.jsp 55
Figura 46 - Interface gráfica da página Sobre.jsp 56

Índice de Tabelas

Tabela 1 – Resumo dos parâmetros 12
Tabela 2 – Áreas de aplicação do ACO 12
Tabela 3 – Parâmetros dos testes 22
Tabela 4 – Resultados dos testes para a sequência 1 22
Tabela 5 – Resultados dos testes para a sequência 2 24
Tabela 6 – Resultados dos testes para a sequência 3 27
Tabela 7 – Resultados dos testes para a sequência 4 31
Tabela 8 – Parâmetros dos testes complementares 34
Tabela 9 – Resultados dos testes com 50 formigas aplicados à sequência 4 35
Tabela 10 – Resultados dos testes com ρ a 0.1, aplicados à sequência 4 36
Tabela 11 – Resultados dos testes com α a 3, aplicados à sequência 4 37
Tabela 12 – Resultados dos testes com β a 3, aplicados à sequência 4 38

Definições e Abreviaturas

ACO	Optimização baseada em colónias de formigas
AS	Ant System
EAS	Elitist Ant System
RAS	Rank-based Ant System
MMAS	Max-Min Ant System
HP	Hydrophobic-Polar
TSP	Travelling Salesman Problem
H	Aminoácido hidrofóbico
P	Aminoácido hidrofílico ou polar
JSP	JavaServer Pages

1 Introdução

1.1 Contexto do Problema

A previsão da estrutura de proteínas é um problema importante e de difícil resolução, que tem vindo a ser estudado por muitos cientistas, desde 1950. Este problema consiste na obtenção da estrutura tridimensional de uma proteína a partir da sua sequência de aminoácidos. A sua resolução pode ser a chave para o conhecimento da origem de certas doenças, e um grande passo para a sua cura.

Uma das possíveis abordagens para resolver este problema é recorrer a simulações computacionais que efectuem a previsão da estrutura tridimensional. O avanço da tecnologia possibilitou a realização destas simulações computacionais, que a nível computacional, são muito exigentes. Dada a complexidade e peso do problema computacional, uma possibilidade é usar modelos HP para simular o *folding*. Estes modelos são muito simples, na medida em que dos vinte aminoácidos originais passamos a ter apenas dois, o H para aminoácidos hidrofóbicos e o P para os hidrofílicos ou polares. Desta forma, uma cadeia de aminoácidos é transformada numa sequência apenas com as letras H e P. A construção é efectuada através da colocação dos aminoácidos numa grelha de duas ou três dimensões, que assume que todos os aminoácidos têm o mesmo tamanho. Esta construção pode ser feita com a ajuda de algoritmos de optimização, guiados por uma função de avaliação que privilegia a obtenção de arranjos com determinadas características. No caso deste trabalho, são utilizados algoritmos baseados em colónias de formigas que são um exemplo de um método baseado em inteligência de enxames.

1.2 Contexto e Objectivos do Projecto

Este projecto foi realizado no âmbito da unidade curricular Estágio/Projecto Industrial do segundo ano de Mestrado em Informática e Sistemas – ramo Desenvolvimento de Software do Instituto Superior de Engenharia de Coimbra.

Os objectivos principais deste projecto foram:

- Conhecer os principais algoritmos de optimização baseados em colónias de formigas.
- Desenvolver e testar algoritmos de optimização baseados em colónias de formigas para resolver problemas de previsão de estruturas de proteínas. As variantes são: o Ant System, Elitist Ant System, Rank-based Ant System e o Max-Min Ant System.
- Desenvolver uma aplicação web baseada em ACO (Optimização baseada em colónias de formigas) para prever a estrutura de proteínas, onde são aplicados os algoritmos implementados. Esta aplicação pode ser utilizada por quem pretenda conhecer e explorar a ferramenta de modo a adquirir conhecimentos sobre os temas abordados, mas também por qualquer pessoa que queira fazer os seus próprios testes, utilizando-a como ferramenta de trabalho ou de estudo.

Com dedicação e trabalho, neste projecto foi conseguido:

- Aprofundar conhecimentos sobre previsão de estrutura de proteínas usando modelos HP.
- Obter conhecimentos sobre algoritmos de inteligência de enxame, mais concretamente ACO's.
- Disponibilizar à comunidade uma aplicação computacional. Esta aplicação permite dar a conhecer os temas abordados, podendo ser utilizada de forma didáctica por quem pretende aprofundar conhecimentos sobre ACO's ou modelos HP. Por outro lado, pode também ser utilizada como ferramenta de trabalho ou de investigação, por pessoas que tenham a intenção de realizar os próprios testes.

1.3 Estratégia de Desenvolvimento

Com base na proposta de projecto, disponível no anexo A, o projecto foi estruturado nas seguintes actividades:

1. Análise e Estudo: aquisição de conhecimentos através de revisão bibliográfica, sobre a previsão da estrutura de proteínas, modelos HP e métodos de optimização baseados em colónias de formigas.

2. Desenvolvimento de algoritmos de optimização: implementação dos algoritmos estudados na fase anterior.
3. Testes: execução de testes aos algoritmos implementados para validação dos mesmos.
4. Concepção e Implementação da aplicação: construção de uma aplicação web, implementação das interfaces e respectivas funcionalidades.

1.4 Estrutura do Relatório

Para além deste primeiro capítulo, este relatório compreende cinco capítulos:

O capítulo 2, Optimização baseada em colónias de formigas, apresenta uma análise detalhada dos modelos computacionais e das variantes de algoritmos baseados em colónias de formigas. Neste capítulo é aprofundado o funcionamento de cada algoritmo.

O capítulo 3, Modelos HP, apresenta uma análise da previsão de estruturas de proteínas e dos modelos HP.

O capítulo 4, Aplicação ACO a modelos HP, apresenta uma descrição de como é aplicado um ACO a um problema HP, e a descrição e análise dos testes aplicados aos algoritmos.

O capítulo 5, Aplicação, apresenta uma descrição do que consiste a aplicação, as suas funcionalidades e interface gráfica.

O capítulo 6, Conclusão, apresenta uma descrição dos resultados do projecto, apreciação crítica e trabalho futuro.

2 Optimização baseada em colónias de formigas

2.1 Introdução

A inteligência de enxame refere-se a sistemas biológicos, constituídos por entidades muito simples, que conseguem colectivamente exibir comportamentos complexos, o que lhes permite resolver problemas com elevado grau de dificuldade. As formigas sempre foram conhecidas como animais trabalhadores e organizados. Os formigueiros conseguem exibir uma vasta gama de comportamentos complexos, entre os quais a descoberta e manutenção de caminhos mais curtos entre dois pontos. Esta capacidade permite-lhes encontrar o melhor modo de criar um caminho até ao alimento e voltar pelo mesmo caminho até ao ninho. Estes agentes comunicam entre si, deixando uma marca no ambiente, a feromona, que é um agente químico. A figura 1 indica o comportamento natural das formigas, de forma a encontrar o caminho mais curto.

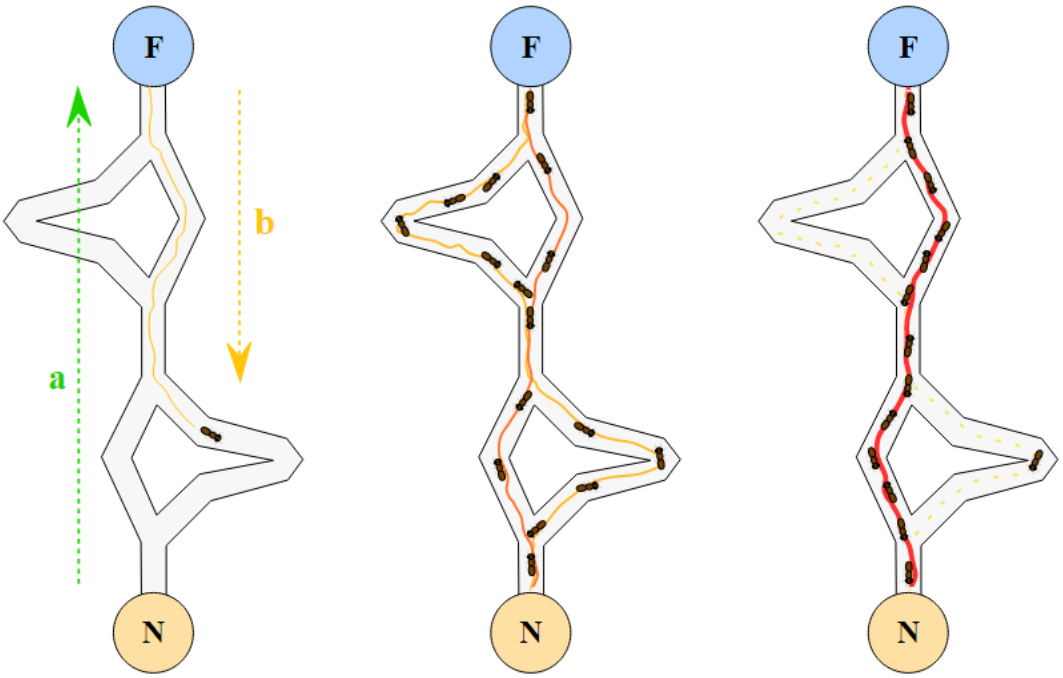

Figura 1 - Comportamento natural das formigas

Na figura 1, existem vários caminhos a ligar o ninho (N) à comida (F), pelos quais as formigas podem passar. No início as formigas começam aleatoriamente a percorrer os vários caminhos e vão depositando feromona, aumentando a probabilidade desse

caminho, ser seleccionado por outra formiga. Depois naturalmente ao longo do tempo a feromona vai-se acumulando, surgindo o trajecto mais curto. À medida que o tempo passa, a feromona deixa de existir nos caminhos menos utilizados pelas formigas. Este fenómeno chama-se evaporação da feromona.

2.2 Modelos computacionais baseados em colónias de formigas

Através da visualização dos comportamentos biológicos e de realizadas experiências, os cientistas inspiraram-se e começaram a criar modelos computacionais, neste caso, criar algoritmos de optimização para resolver problemas complexos. A optimização baseada em colónias de formigas (ACO) é uma metaheurística que se baseia numa colónia de formigas artificial, que têm o objectivo de encontrar soluções para problemas de optimização difíceis.

Para resolver um problema utilizando um ACO, é necessário modelá-lo como um grafo. Um grafo G = (V,A) é definido por um conjunto, de vértices V, correspondentes às componentes da solução, e por um conjunto de arestas A, que são as ligações entre as componentes.

As formigas artificiais vão resolver o problema, percorrendo o grafo e recolhendo componentes para construir uma solução. A travessia e selecção das componentes vão ser realizadas, tendo em atenção eventuais restrições associadas ao problema e utilizando informação existente nas arestas: as feromonas e a heurística. A feromona é uma medida que indica até que ponto a ligação surgiu em boas soluções anteriormente. A heurística é uma medida específica ao problema que está a ser resolvido, e indica se essa ligação tenderá a fazer parte ou não de boas soluções. Depois de construída, a qualidade da solução é avaliada pela função de avaliação associada ao problema [42][43]. A figura 2 é um exemplo da representação de um problema como um grafo.

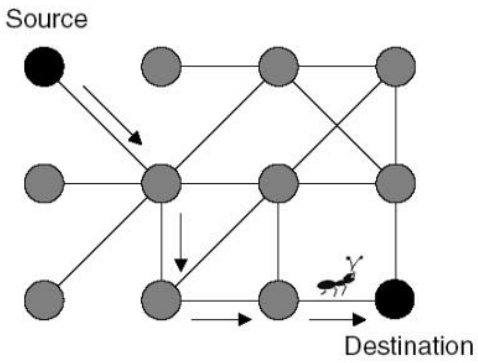

Figura 2 – Exemplo da representação de um grafo

Por exemplo, vamos assumir que queremos resolver o muito conhecido problema do TSP (Problema do caixeiro viajante). Neste problema, é considerado um conjunto de cidades (vértices do grafo) e as ligações entre elas (arestas do grafo). O objectivo é encontrar o caminho mais curto, de modo que cada cidade seja visitada mas apenas uma vez [42][44]. As restrições deste problema são que todas as cidades devem ser visitadas (vértices) e cada cidade apenas pode ser visitada uma vez. Inicialmente é escolhida uma cidade aleatoriamente, como é indicado na figura 3.

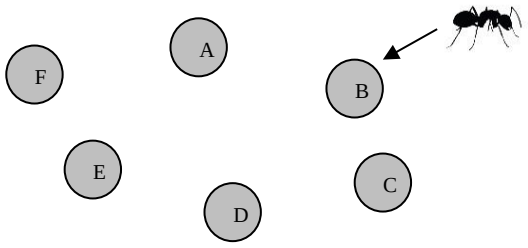

Figura 3 – Escolha aleatória da cidade

De seguida a formiga calcula o potencial da ligação para cada cidade ainda não visitada, com as informações das feromonas e da heurística, como é mostrado na figura 4. A feromona reflecte até que ponto a ligação entre duas cidades específicas, tem sido muito usada em boas soluções anteriormente. A heurística é o inverso da distância entre as cidades.

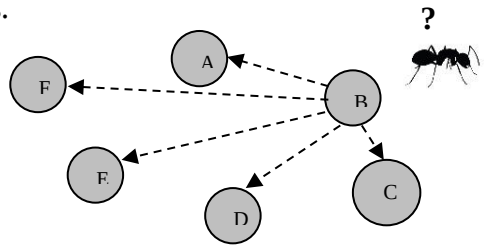

Figura 4 – Cálculo do potencial de cada ligação

Este cálculo serve para escolher a ligação mais promissora. A escolha probabilística continua até chegar à última cidade. Uma possível solução é apresentada na figura 5.

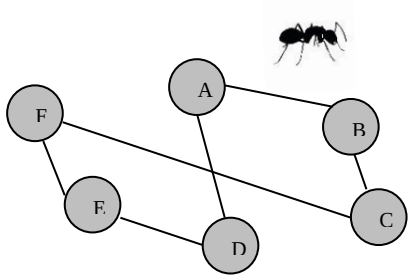

Figura 5 – Uma possível solução para o TSP

Para uma melhor compreensão deste tipo de resolução de problemas, vai ser apresentado o pseudocódigo do algoritmo geral de ACO:

Algoritmo ACO
Definição de parâmetros
Inicialização de feromonas
Enquanto a condição de paragem não for atingida **repetir**:
Construção de Soluções
Avaliação de Soluções
Actualização de Feromonas
Fim Enquanto
Devolve a melhor solução

No pseudocódigo são apresentados três procedimentos principais:

1. Construção de Soluções consiste na construção das soluções pelas formigas, ou seja cada formiga visita as várias componentes, e das várias ligações possíveis, esta vai escolher a mais promissora para avançar para a próxima componente, e assim sucessivamente até finalizar a construção. O potencial da ligação, que se baseia numa função probabilística, vai depender do valor das feromonas e do valor da heurística.

2. Avaliação de Soluções, consiste em avaliar todas as soluções de acordo com a função de avaliação associada ao problema em questão.

3. Actualização de Feromonas consiste em actualizar as feromonas através das operações de evaporação e do reforço. Na evaporação, a feromona de todas as arestas diminui ligeiramente. De seguida é feito o reforço, em que apenas as ligações usadas pelas formigas na solução, são reforçadas. A quantidade de

feromona depositada por cada formiga é directamente proporcional à qualidade da sua solução. O reforço das feromonas aumenta a probabilidade dessas arestas serem escolhidas nas próximas iterações. A evaporação faz com que novas ligações sejam exploradas e evita que o algoritmo tenha a tendência para convergir muito rapidamente para um trajecto que seja um óptimo local.

Todo este processo repete-se, até atingir o número de iterações pretendido. [46]

2.3 Variantes de algoritmos baseados em colónias de formigas

Existem diversas variantes de algoritmos baseados em colónias de formigas. Estas variantes seguem o algoritmo geral que foi explicado anteriormente, tendo cada variante as suas alterações e melhorias. Neste subcapítulo vão ser descritas as diferenças que estes algoritmos têm no que diz respeito à construção de soluções e na utilização das feromonas.

2.3.1 Ant System

O Ant System foi apresentado por Marco Dorigo, em 1992. Este algoritmo passa por diversas fases, a inicialização das feromonas, a construção das soluções e avaliação das mesmas, evaporação e reforço das feromonas.

Inicialmente, antes das formigas começarem a fase de construção das soluções, todas as ligações são inicializadas com o mesmo valor de feromona:

$$\tau_0 = m/C^{nn} \qquad (1)$$

O m é o número de formigas e C^{nn} é a qualidade de uma solução gerada por uma heurística sôfrega.

As formigas, quando estão na fase de construção, utilizam os valores das feromonas e da heurística para decidir qual a ligação escolhida. A fórmula utilizada para o cálculo é apresentada em (2):

$$p_{ij} = \frac{[\tau_{ij}]^{\alpha}[\eta_{ij}]^{\beta}}{\Sigma_{l \in N_i^k}[\tau_{il}]^{\alpha}[\eta_{il}]^{\beta}}, se\ j \in N_i \qquad (2)$$

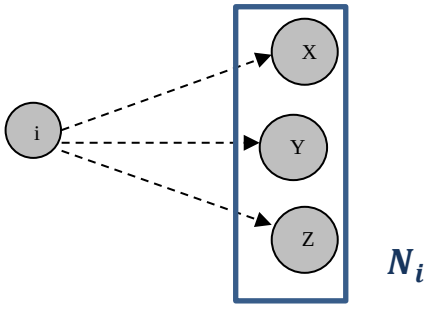

Figura 6 – Ligações possíveis a partir do vértice i

Na figura 6, a formiga i está localizada no vértice i e o $\boldsymbol{N_i}$ é o conjunto dos vértices para onde se pode deslocar. Na figura 6 exemplifica-se uma situação em que existem três ligações possíveis, os vértices X, Y e Z. A probabilidade da ligação (i,j) é composta por dois elementos:

- $[\tau_{ij}]^{\alpha}$, onde τ_{ij} é a feromona presente na aresta (i,j) e o α um peso atribuído.
- $[\eta_{ij}]^{\beta}$, onde η_{ij} é o valor da heurística (definido à priori) na aresta (i,j) e o β um peso atribuído.

O divisor que surge em (2), é um factor de normalização. Os valores α e β são parâmetros do algoritmo e vão determinar o peso relativo que a feromona e a heurística vão ter no cálculo da probabilidade. Após serem calculadas as probabilidades, a escolha de ligações é feita de forma estocástica.

Após a construção das soluções, as feromonas são actualizadas. A actualização envolve a evaporação e o reforço das feromonas. O primeiro processo é a evaporação, que é feita de acordo com a seguinte fórmula (3):

$$\tau_{ij} \leftarrow (1-\rho)\tau_{ij}, \quad \forall (i,j) \in L \qquad (3)$$

O ρ é a taxa de evaporação e varia entre 0 e 1. O parâmetro ρ é muito importante pois evita a acumulação ilimitada da feromona em todas as arestas do grafo. De seguida é aplicado o reforço, segundo a fórmula:

$$\tau_{ij} \leftarrow \tau_{ij} + \sum_{k=1}^{m} \Delta\tau_{ij}^{k}, \quad \forall (i,j) \in L \qquad (4)$$

A fórmula indica que à feromona disponível na aresta, é adicionada a contribuição de cada uma das *m* formigas. $\Delta\tau_{ij}^{k}$ é a contribuição da formiga k.

$$\Delta\tau_{ij}^{k} = \begin{cases} 1/C^{k} \\ 0 \end{cases} \qquad (5)$$

Se tiver passado nessa aresta, vai reforçar a feromona com uma quantidade directamente proporcional à qualidade da sua solução C^k. Se a formiga k não passou na ligação, a sua contribuição é 0.

2.3.2 Elitist Ant System

O Elitist Ant System foi uma tentativa de melhorar o Ant System e a alteração introduzida foi que a melhor solução tem um reforço adicional. A forma como são construídas as soluções, e a evaporação das feromonas é realizada da mesma forma como no AS. A fórmula (6) usada para o reforço é a seguinte:

$$\tau_{ij} \leftarrow \tau_{ij} + \sum_{k=1}^{m} \Delta\tau_{ij}^{k} + e\Delta\tau_{ij}^{bs} \quad (6)$$

A parte $e\Delta\tau_{ij}^{bs}$ corresponde à alteração introduzida, em que o e é o peso dado pela melhor solução até ao momento e $\Delta\tau_{ij}^{bs}$ é contribuição da melhor formiga no reforço das feromonas.

2.3.3 Rank-based Ant System

O Rank-based Ant System é outra variante do EAS e do AS e as principais alterações em relação ao AS são: apenas as melhores soluções é que reforçam e a melhor de todas efectua um reforço adicional.

Antes do reforço, as soluções são ordenadas por qualidade e apenas as que estão melhor colocadas é que são seleccionadas para reforçar. A fórmula do reforço é a seguinte:

$$\tau_{ij} \leftarrow \tau_{ij} + \sum_{r=1}^{w-1}(w-r)\Delta\tau_{ij}^{r} + w\Delta\tau_{ij}^{bs} \quad (7)$$

O número de formigas seleccionadas w é um parâmetro do algoritmo.

Se assumirmos que w=5 então temos:

- Primeira formiga r=1, na fórmula fica (5-1) $\Delta\tau_{ij}^{r}$, ou seja a primeira formiga vai reforçar 4 vezes.
- Segunda formiga r=2, na fórmula fica (5-2) $\Delta\tau_{ij}^{r}$, ou seja a segunda formiga vai reforçar 3 vezes.

E assim sucessivamente até todas as formigas seleccionadas efectuarem o seu reforço.

A parte da fórmula $w\Delta\tau_{ij}^{bs}$ indica que a melhor solução é que vai reforçar novamente, assim como acontece no EAS.

2.3.4 Max-Min Ant System

O Max-Min AS introduz quatro modificações em relação ao AS: apenas uma formiga reforça as feromonas tornando a pesquisa muito sôfrega. Para contrabalançar isto, a inicialização das feromonas é feita de forma diferente. Além disso, são também introduzidos limites que as feromonas não podem ultrapassar e é criado um mecanismo de reinicialização das feromonas.

A inicialização das feromonas é feita com um valor elevado (t_{max}). A reinicialização das feromonas é feita quando o algoritmo estagna, e por isso a reinicialização vai evitar a convergência. A reinicialização acontece quando o algoritmo detecta que não existem melhorias nas soluções num determinado número de iterações sucessivas.

O reforço é realizado de acordo com a fórmula seguinte:

$$\tau_{ij} \leftarrow \tau_{ij} + \Delta\tau_{ij}^{best} \quad (8)$$

A fórmula indicada em (8), mostra que apenas a melhor solução até ao momento é que é que vai reforçar.

Os limites das feromonas τ_{min} e τ_{max} são sempre calculados quando é encontrada uma nova melhor solução. As fórmulas são as seguintes:

$$t_{max} = 1/\rho C^{bs} \quad (9)$$

$$\tau_{min} = t_{max}\left(1 - \sqrt[n]{0.05}\right)/\left((avg - 1).\sqrt[n]{0.005}\right) \quad (10)$$

Detalhando a fórmula (9), o parâmetro ρ é a taxa de evaporação e C^{bs} é o custo da melhor solução encontrada até ao momento. O parâmetro avg da fórmula (10) é o número médio de ligações e o *n* é o tamanho do grafo.

Os limites são aplicados no reforço das feromonas, ou seja, sempre que uma feromona é actualizada, o algoritmo vai verificar se a feromona se encontra dentro dos limites. Se o valor da feromona estiver acima do t_{max}, a feromona vai ficar com valor igual a t_{max}. Se o valor da feromona estiver abaixo do τ_{min}, a feromona vai ficar com valor igual a τ_{min}.

2.4 Parâmetros

A tabela 1 mostra um resumo dos parâmetros que são necessários especificar para aplicar um ACO na resolução de problemas. Nesta tabela, são também indicados alguns valores padrão normalmente utilizados [46].

Tabela 1 – Resumo dos parâmetros

Parâmetros	Valores padrão
Número de formigas	Número de vértices do grafo
Número de iterações	Depende do problema
α	1,2,3
β	1,2,3,4,5
ρ	0.02 a 0.5
w	6

2.5 Áreas de aplicação

Estes métodos têm sido aplicados ao longo destes últimos anos em diferentes problemas de optimização, das mais diferentes áreas com bons resultados. Neste artigo [29] é feita uma listagem de vários tipos de problemas e variantes de aplicação. A tabela 2 apresenta um resumo de alguns dos principais problemas abordados.

Tabela 2 – Áreas de aplicação do ACO

Tipo de Problema	Nome do Problema	Referência
Routing	Traveling salesman	[1], [2], [3], [4], [5]
	Vehicle routing	[6], [7]
	Sequencial ordering	[8]
Assignment	Quadratic assignment	[4], [9]
	Course timetabling	[10], [11]
	Graph coloring	[12]

Scheduling	Project scheduling	[13]
	Total weighted tardiness	[14]
	Total weighted tardiness	[15]
	Open shop	[16]
Subset	Set covering	[17]
	l-cardinality trees	[18]
	Multiple knapsack	[19]
	Maximum clique	[20]
Other	Constraint satisfaction	[21], [22]
	Classification rules	[23], [24]
	Bayesian networks	[25], [26]
	Protein folding	[27]
	docking	[28]

3 Modelos HP

3.1 Introdução

As proteínas são "colares" de aminoácidos, ou seja, moléculas de cadeia longa e são a base da biologia. Estas são essenciais para o nosso organismo, pois cada proteína tem a sua função. As principais funções das proteínas são o transporte, estrutura, catalisação, regulação, controlo e resposta imunitária. Para as proteínas executarem as suas funções correctamente, devem estar na estrutura terciária, ou chamada estrutura nativa, e para isso passam pelo processo de transcrição e tradução do DNA como é mostrado na figura 7. [33]

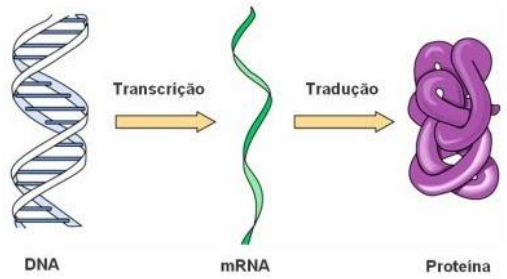

Figura 7- Esquema da Transcrição e Tradução do DNA

O DNA é como se fosse a receita para a construção das proteínas, ou seja, especifica a sequência de aminoácidos dispostos ao longo do "colar" de proteínas. Essencialmente o que ocorre na transcrição, é a transferência da informação genética do DNA para RNA e na tradução é realizada a descodificação do RNA em proteína. [34][35]

As proteínas têm vários níveis estruturais:

- Estrutura primária: Sequência de aminoácidos que compõe a cadeia polipeptídica.
- Estrutura secundária: Contém estruturas locais e repetitivas onde se destacam as folha-beta e alfa-hélice.
- Estrutura Terciária: organização tridimensional das estruturas secundárias numa cadeia polipeptídica.

A figura 8 ilustra os diferentes níveis estruturais das proteínas. [36] [37] [39]

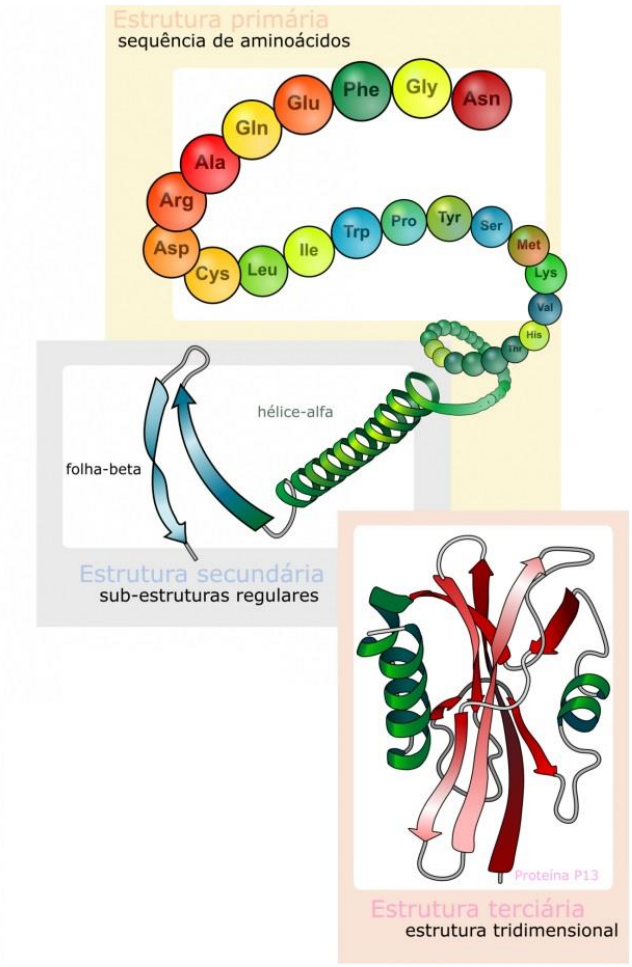

Figura 8- Nível estrutural das proteínas

O problema principal abordado neste projecto, a previsão de estruturas de proteínas consiste em prever, a partir da estrutura primária, a estrutura tridimensional. A sequência de aminoácidos passa pelo processo de *folding* e originará a estrutura tridimensional da proteína. O *folding* é um processo complexo que envolve o enrolamento e dobragem da cadeia de aminoácidos. O problema pode ser definido como: dada uma sequência de aminoácidos, como obtemos a estrutura tridimensional? O objectivo é prever como o *folding* é feito. Actualmente ainda não existe nenhum algoritmo, nem solução que resolva de forma precisa este problema.

As proteínas antes de executarem as suas funções assumem rapidamente a sua forma tridimensional. O bom funcionamento do organismo depende muito da existência e funcionamento correcto das proteínas. Os diabetes do tipo I e a hemofilia, ocorrem visto que o organismo encontra-se incapaz de produzir insulina e o factor VIII. As doenças como Alzheimer, fibrose cística, BSE (doença das vacas loucas) apontam como origem o processo de *folding* defeituoso de que resultam proteínas, ou agentes

infecciosos que provocam essas doenças. Acredita-se que a solução era sermos capazes de fornecer a proteína em falta ou até mesmo inactivar uma proteína infecciosa, mas para isso é necessário conhecer e compreender como se processa o *folding* de proteínas.

Christian Anfinsen, um cientista americano de origem norueguesa, foi dos primeiros cientistas a investigar sobre o processo *folding*, durante a década de 1950, pelo que em 1972 recebeu o Prémio Nobel da Química. A sua investigação baseou-se em duas questões: por que razão se dobra a proteína para a estrutura nativa? Por que é única essa estrutura? Estas questões levaram à elaboração da hipótese termodinâmica que indica que o processo *folding* ocorre de forma espontânea conduzindo ao estado nativo, sendo o mais estável (de mais baixa energia) do ponto de vista termodinâmico. Esta experiência foi o ponto de partida para o problema actual de determinar a estrutura terciária a partir da sequência de aminoácidos.

Nos anos 90 do século XX, devido ao aumento da capacidade de cálculos pelos computadores, começou-se a fazer simulações computacionais por dinâmica molecular (DM) para estudar o *folding*. A dinâmica molecular, neste caso baseia-se no conhecimento das interacções que se estabelecem entre os átomos de uma proteína. Estas simulações são extremamente exigentes a nível computacional. É necessário um dia de CPU para processar um nanossegundo do processo *folding*, sendo que o *folding* de uma proteína razoável demora 10000ns. Por estes motivos, é possível aplicar uma abordagem mais simples como por exemplo o modelo *Hydrophobic-Polar* (HP), que vai ser explicado em detalhe no ponto 3.2. [38][39]

Alguns projectos *folding* são o "Folding@Home" e "Protein modeling with Blue Gene/L" que podem ser vistos nas referências [47] e [48] respectivamente.

3.2 Modelos HP

O modelo HP (*Hydrophobic-Polar)* é um modelo simples que apenas considera aminoácidos hidrofóbicos (H) e polares (P), e permite analisar as interacções entre os aminoácidos e as moléculas de água durante o processo de *folding*. Basicamente, a cadeia de aminoácidos é transformada numa sequência de aminoácidos só com as letras H e P e depois é aplicado o modelo HP para simular o *folding*. A construção da

estrutura da proteína, apenas com os aminoácidos H e P, pode ser feita numa grelha bidimensional ou tridimensional. A configuração da grelha é variável e cada célula pode estar vazia ou conter um aminoácido. Os aminoácidos que estão adjacentes na sequência devem-se manter adjacentes na grelha e não pode haver sobreposição de aminoácidos. Dada a sequência HHPHPPHH, a construção é mostrada na figura 9.

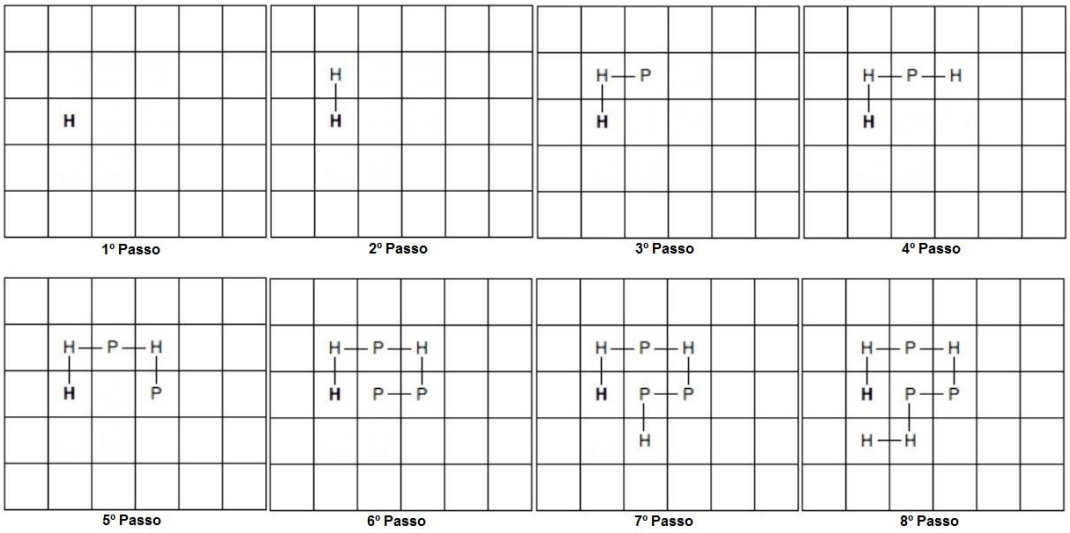

Figura 9 – Passos de construção de uma solução

O objectivo é encontrar a distribuição ideal, que possa obter o maior número de ligações H-H adjacentes na grelha visto que a qualidade de uma solução (disposição final dos aminoácidos na grelha), é dada pela seguinte fórmula:

$$Qualidade(S) = \sum contactos(H-H) \times \varepsilon \qquad (10)$$

A qualidade é igual ao número de ligações H-H adjacentes na grelha, multiplicado pela energia de interacção dos aminoácidos hidrofóbicos que é -1, a energia entre outros aminoácidos é considerada 0. A energia dos aminoácidos hidrofóbicos é mais baixa, o que vai fazer com que estes aminoácidos tenham a tendência de se juntarem mais no núcleo, ficando os aminoácidos hidrofílicos no exterior, maximizando as interacções entre si. [40] [41]Para a sequência mostrada anteriormente, onde a construção é mostrada na figura 9, duas possíveis soluções são mostradas na figura10.

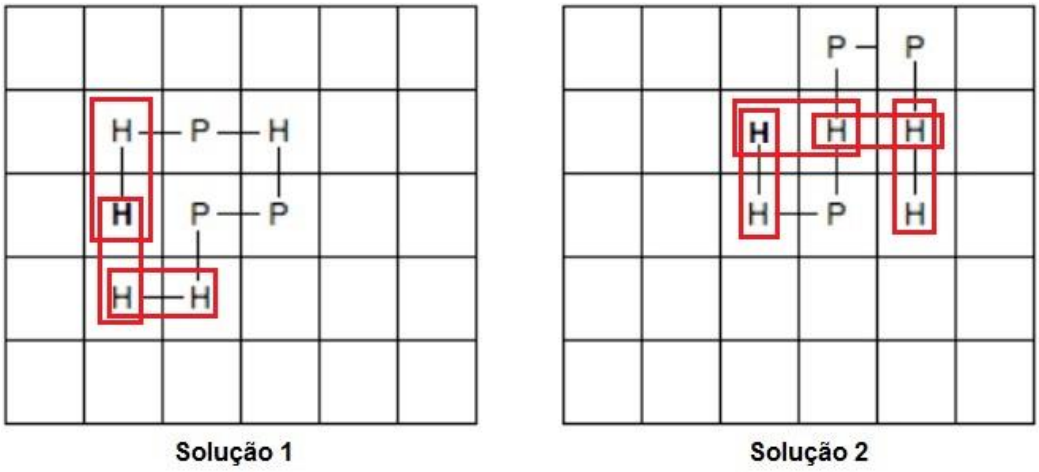

Figura 10 – Exemplo de soluções de uma determinada sequência

Na solução 1 podemos verificar 3 ligações H-H e por isso tem qualidade -3. A solução 2 tem 4 ligações H-H e a sua qualidade é -4. A solução 2 é melhor, visto que tem um maior número de ligações.

Este é um problema de optimização NP, pelo que não é possível encontrar soluções óptimas em tempo útil. Por esse motivo, recorre-se normalmente a algoritmos de aproximação para encontrar soluções de boa qualidade num tempo aceitável.

Um modelo é uma simplificação e isso traz vantagens e desvantagens. As vantagens têm a ver com o facto de o problema ser possível de resolver, as desvantagens tem a ver com o facto de nos afastarmos do problema real. Os modelos HP apresentam algumas vantagens importantes, como ser um modelo simples e sem parâmetros, que permitem fazer uma aproximação da previsão da estrutura de proteínas. A introdução de simplificações neste modelo leva a algumas limitações como: só tem dois tipos de aminoácidos (H e P), têm todos o mesmo tamanho.

3.3 Outras abordagens para a resolução de modelos HP

A resolução de modelos HP é um tema que tem sido abordado em diversos trabalhos. A seguir descreve-se brevemente três dessas abordagens:

- "*CPSP-web-tools: a server for 3D lattice protein studies*", a ferramenta Constraint-based protein structure prediction (CPSP) é um pacote que contém algoritmos muito rápidos para previsão de estruturas ab initio e problemas

relacionados em modelos HP 3D. CPSP-web-tools é uma ferramenta interactiva online da ferramenta CPSP [30].

- "*An ant colony optimization algorithm for the 2D and 3D hydrophobic polar protein folding problem*". Neste trabalho, foi introduzido um algoritmo ACO, para resolver um problema combinatório NP-difícil para prever a estrutura da proteína a partir da sequência de aminoácidos com modelos HP 2D e 3D [31].
- "*Search strategies in structural bioinformatics*". Este trabalho fala sobre alguns desafios de bioinformática, como:
    - Investigação sobre comparação da estrutura da proteína, e realçar a utilidade da complexidade de Kolmogorov como uma medida de similaridade estrutural.
    - Discussão sobre alguns desenvolvimentos na previsão de estruturas de proteínas, como a previsão da estrutura off-lattice usando o algoritmo do dilúvio.
    - Destacam os algoritmos memetic, que combinam algoritmos genéticos com optimização local, para estudar modelos simples de proteínas em grelhas de duas e três dimensões [32].

4 Aplicação de ACO a modelos HP

4.1 Como aplicar um ACO a um problema HP

Este subcapítulo tem como objectivo descrever a aplicação de um ACO a um problema HP. Dada uma sequência HP, constituída por aminoácidos H e P, pretende-se simular o seu *folding*, numa grelha bidimensional. Um ACO vai resolver este problema de optimização. Neste tipo de algoritmos, a modelação do problema é feita com um grafo, que corresponde à grelha de duas dimensões nos modelos HP. Os vértices são as células da grelha e as arestas assinalam as adjacências das células.

A construção das soluções passa por inicialmente colocar na célula central o primeiro símbolo. Enquanto não acabar a sequência, a formiga vai colocando o próximo símbolo numa célula adjacente livre, até colocar todos os símbolos. Depois de construída a solução, é feita a avaliação aplicando a fórmula representada em (10). Existe também a possibilidade, de a formiga chegar a um beco sem saída, e a construção não poder prosseguir, sendo por isso considerada uma solução inválida. Por exemplo, dada a sequência HHPHPPHHPHP, na figura 11 demonstra dois passos que levaram a uma solução inválida.

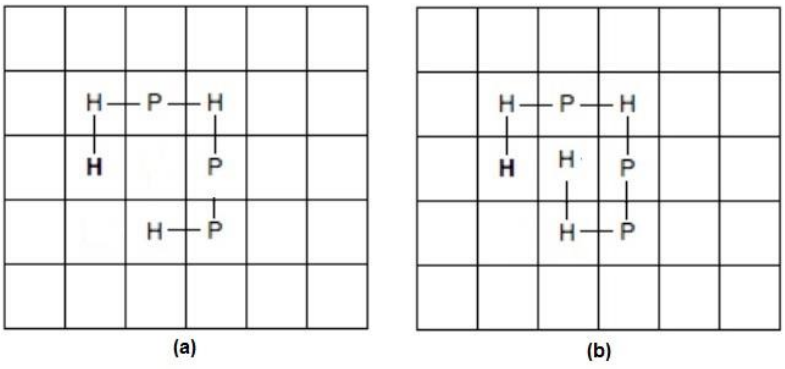

Figura 11 – Solução inválida

Na grelha (a), a formiga tinha três opções, das quais duas eram promissoras, e a formiga foi escolher a célula que levou a uma solução inválida como é mostrado em (b), ficando os aminoácidos PHP fora da construção. A avaliação das soluções inválidas consiste numa penalização, e esta é feita em função do número de símbolos não colocados. A penalização é directamente proporcional à quantidade de símbolos por colocar.

Depois da avaliação, é realizada a actualização das feromonas. As feromonas são actualizadas de acordo com o preenchimento das células nesta iteração.

A heurística consiste em calcular o número de vizinhos das células, que estão à volta da célula onde a formiga se encontra.

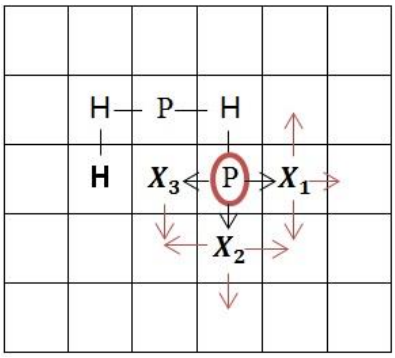

Figura 12 - Heurística

A figura 12 exemplifica a aplicação da heurística e podemos verificar que se a formiga estiver na célula com aminoácido P assinalado, esta tem três células livres que poderá escolher. O objectivo da heurística é fazer com que a formiga evite ficar "encurralada" e escolher as células com maior número de opções. Neste caso do aminoácido seleccionado, as células $\boldsymbol{X_1}$ e $\boldsymbol{X_2}$ não têm qualquer vizinho ocupado, tornando-as boas opções para a formiga escolher, ao contrário da célula $\boldsymbol{X_3}$ que já tem dois vizinhos ocupados.

4.2 Testes

Neste subcapítulo pretende-se explicar como foram realizados os testes e os parâmetros definidos. Para a realização destes testes, foram escolhidas quatro sequências de tamanhos diferentes, números específicos de formigas e iterações, e heurística. A tabela 3 mostra os parâmetros utilizados nos testes. As sequências utilizadas são as seguintes:

1. HHPHHPPHPHH
2. HPHPPHHPHPPHPHHPPHPH
3. PPHPPHHPPHHPPPPPHHHHHHHHHHPPPPPPHHPPHHPPHPPHHHHH

4. HHHHHHHHHHHHPHPHPPHHPPHHPPHPPHHPPHHPPHPPHHPPHHPPHP
   HPHHHHHHHHHHHH

O método de optimização está parametrizado com as configurações indicadas na tabela 3.

Tabela 3 – Parâmetros dos testes

Algoritmos	AS, EAS, RAS, MMAS
Sequências	1, 2, 3, 4
Heurística (H)	Sim (S), Não (N)
Número de Formigas	10
Número de iterações (Nº it)	10, 50, 100
α	1
β	1
ρ	0.5
w	5
Número de repetições	1

Nas tabelas com os resultados é apresentada a seguinte informação: percentagem de inválidos (%), melhor solução da última iteração (Qualidade), média das soluções da última iteração (Média) e a melhor solução encontrada pelo algoritmo (MS).

4.2.1 Resultados dos Testes aplicados à Sequência 1

Tabela 4 – Resultados dos testes para a sequência 1

	H	Nº it	%	Qualidade	Média	MS
AS	N	10	0	-18	-15	-18
		50	0	-18	-16	-19
		100	10	-18	-16	-19

	S	10	0	-18	-16	-18
		50	0	-18	-15	-18
		100	0	-17	-16	-18
EAS	N	10	10	-18	-15	-18
		50	10	-17	-15	-19
		100	0	-18	-16	-19
	S	10	0	-18	-16	-18
		50	0	-18	-16	-18
		100	0	-17	-15	-18
RAS	N	10	0	-18	-16	-19
		50	0	-18	-17	-19
		100	0	-18	-16	-19
	S	10	0	-18	-16	-18
		50	0	-18	-16	-18
		100	0	-17	-16	-18
Max-Min	N	10	0	-18	-15	-18
		50	10	-16	-14	-19
		100	0	-18	-15	-19
	S	10	0	-17	-14	-18
		50	0	-18	-16	-18
		100	0	-18	-15	-18

Relativamente aos resultados descritos na tabela 4, estes não diferem muito de uns algoritmos para os outros. Os valores da qualidade e média da última iteração, e a melhor solução não variam muito com a aplicação dos diferentes algoritmos. Em relação à aplicação da heurística, podemos verificar que em alguns algoritmos ajudou

na obtenção de melhores resultados. Em alguns casos, apesar de poucos, sem heurística, a percentagem de inválidos chegou aos 10%, e com a heurística, todos os algoritmos tiveram resultados de 0% de percentagem de inválidos.

4.2.2 Resultados dos Testes aplicados à Sequência 2

Tabela 5 – Resultados dos testes para a sequência 2

	H	Nº it	%	Qualidade	Média	MS
AS	N	10	10	-26	-23	-26
		50	20	-26	-22	-29
		100	30	-26	-22	-29
	S	10	0	-26	-23	-26
		50	0	-25	-23	-27
		100	0	-25	-23	-27
EAS	N	10	20	-26	-21	-27
		50	0	-27	-24	-29
		100	10	-27	-23	-29
	S	10	0	-26	-23	-26
		50	0	-25	-23	-27
		100	0	-25	-23	-28
RAS	N	10	0	-25	-24	-27
		50	20	-27	-22	-29
		100	20	-27	-22	-28
	S	10	0	-26	-23	-27
		50	0	-26	-24	-27
		100	10	-27	-23	-27
Max-	N	10	0	-24	-23	-26

Min		50	10	-26	-22	-29
		100	20	-26	-21	-28
	S	10	0	-25	-23	-26
		50	0	-26	-23	-27
		100	0	-24	-23	-26

Relativamente aos resultados obtidos na tabela 5, a qualidade, a média e a melhor solução não variam muito de uns algoritmos para os outros. A percentagem de inválidos demonstra que a heurística ajudou na obtenção de melhores resultados.

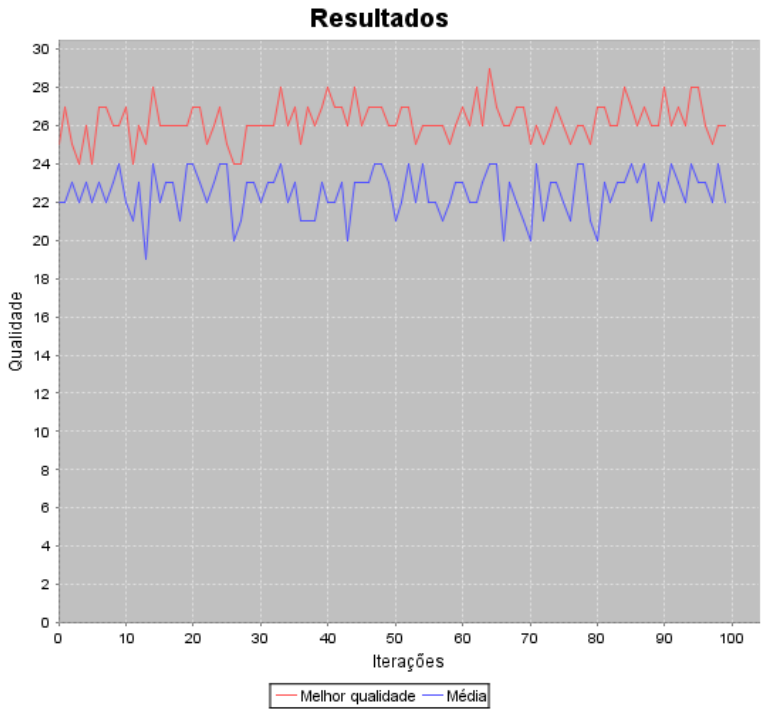

Figura 13 - Resultados obtidos da aplicação do AS sem heurística com 100 iterações

Figura 14 - Resultados obtidos da aplicação do AS com heurística com 100 iterações

De modo a complementar os dados da tabela 5, as figuras 13 e 14 apresentam a média e a melhor qualidade por iteração. A figura 13 apresenta a média mais inconstante e praticamente variando de iteração para iteração, e muitas vezes com picos acentuados. Os valores da média da figura 14 são mais constantes, e não existem variações muito bruscas de umas iterações para as outras.

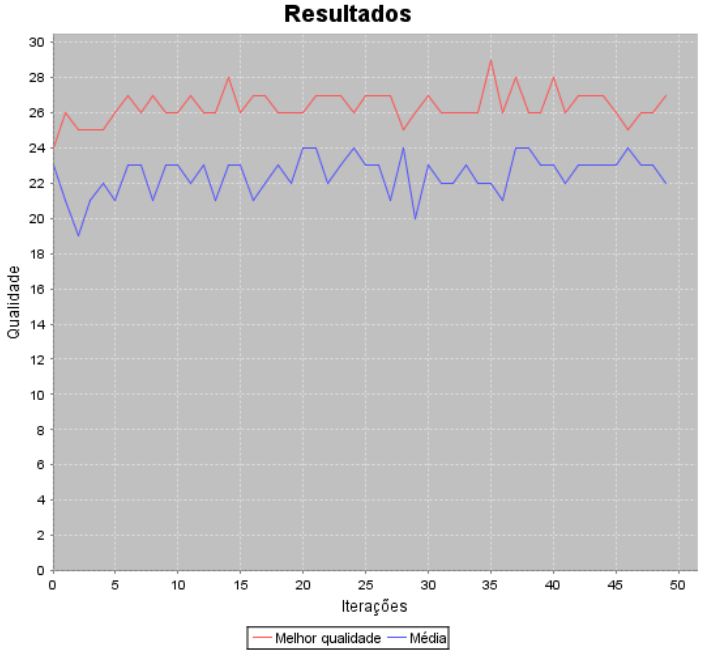

Figura 15 - Resultados obtidos da aplicação do RAS sem heurística com 50 iterações

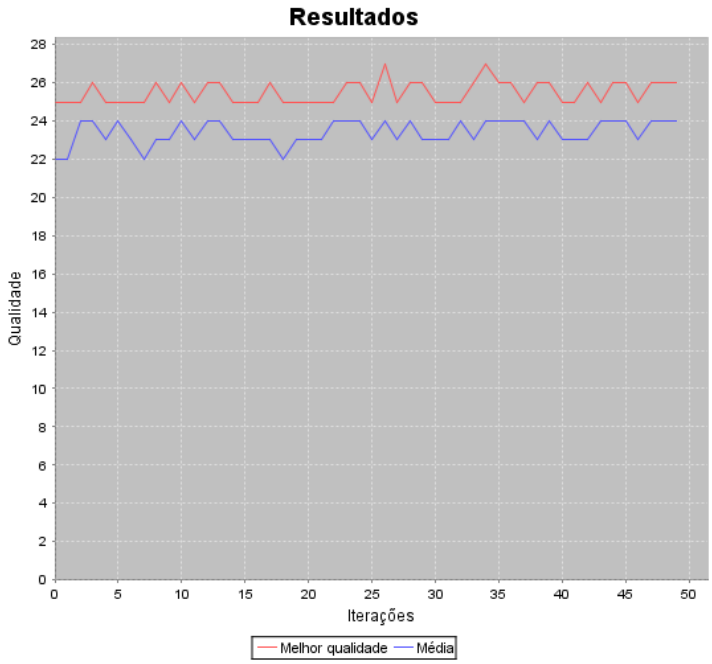

Figura 16- Resultados obtidos da aplicação do RAS com heurística com 50 iterações

As figuras 15 e 16, à semelhança dos resultados obtidos nas figuras 13 e 14, mas com a aplicação de um algoritmo diferente, o RAS, verifica-se um comportamento mais contante da média, quando a heurística é aplicada (figura 16).

4.2.3 Resultados dos Testes aplicados à Sequência 3

Tabela 6 – Resultados dos testes para a sequência 3

	H	Nº it	%	Qualidade	Média	MS
AS	N	10	50	-74	-46	-75
		50	50	-71	-50	-76
		100	40	-70	-46	-77
	S	10	30	-70	-58	-73
		50	30	-72	-59	-73
		100	50	-72	-52	-76
EAS	N	10	50	-75	-47	-76
		50	70	-71	-38	-77

		100	50	-72	-50	-76
	S	10	30	-70	-58	-74
		50	20	-70	-62	-75
		100	20	-72	-62	-75
RAS	N	10	50	-73	-50	-73
		50	40	-71	-55	-74
		100	40	-72	-51	-75
	S	10	50	-73	-50	-73
		50	40	-71	-55	-74
		100	40	-72	-51	-75
Max-Min	N	10	40	-71	-54	-76
		50	30	-70	-56	-77
		100	20	-70	-59	-77
	S	10	40	-69	-54	-73
		50	10	-69	-63	-74
		100	10	-69	-63	-75

Os resultados obtidos na tabela 6, relativamente à sequência 3, revelam que os valores da qualidade e da melhor solução não variam muito de uns algoritmos para os outros. Os valores da média apesar de também não variarem muito, estão relacionados com a percentagem de inválidos, quanto maior essa percentagem, mais baixa é a média.

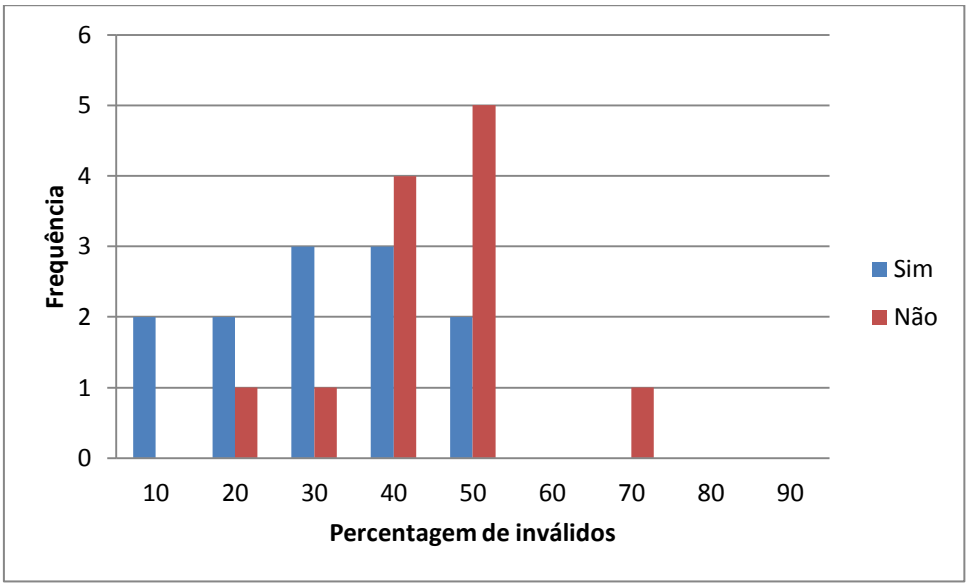

Figura 17 – Percentagem de inválidos na última iteração de experiências com e sem heurística

A figura 17 pretende resumir e demonstrar de forma mais perceptível a relação entre a heurística e percentagem de inválidos. As barras a azul dizem respeito aos resultados dos testes com heurística e as barras a vermelho aos resultados dos testes sem heurística, indicados na tabela 6. Podemos tirar uma primeira conclusão de que, o tamanho da sequência influencia numa melhor aplicação da heurística, visto que nos resultados da sequência 2, ver no ponto 4.2.2, as percentagens de inválidos não eram tão elevadas.

Relativamente aos dados da figura 17, apesar da aplicação da heurística não ter sido tão eficaz na obtenção de melhores resultados, a percentagem de inválidos nunca foi superior aos 50%. Na maior parte das experiências sem heurística, a percentagem de inválidos está entre os 40% e os 50%. Não sendo estes resultados, tabela 6 e figura 17, resultados absolutos, a análise de outro tipo de resultados é importante.

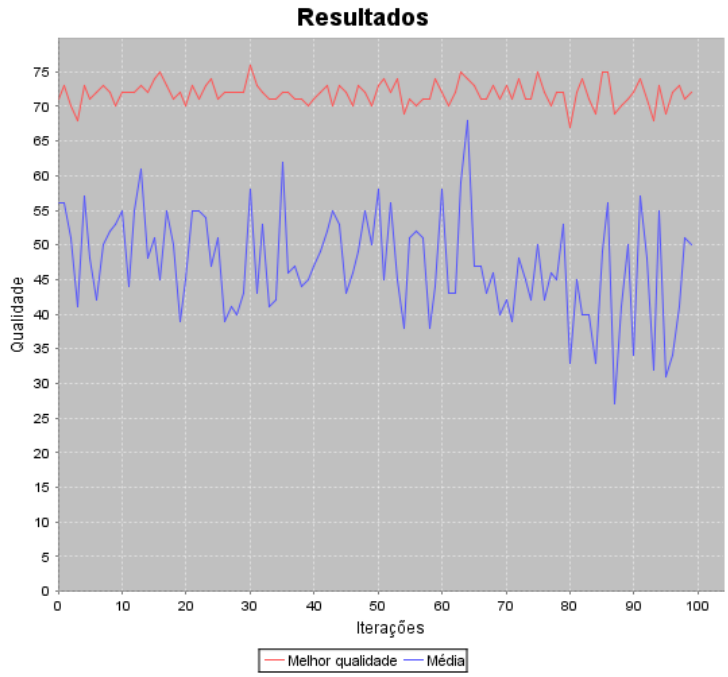

Figura 18 - Resultados obtidos da aplicação do EAS sem heurística com 100 iterações

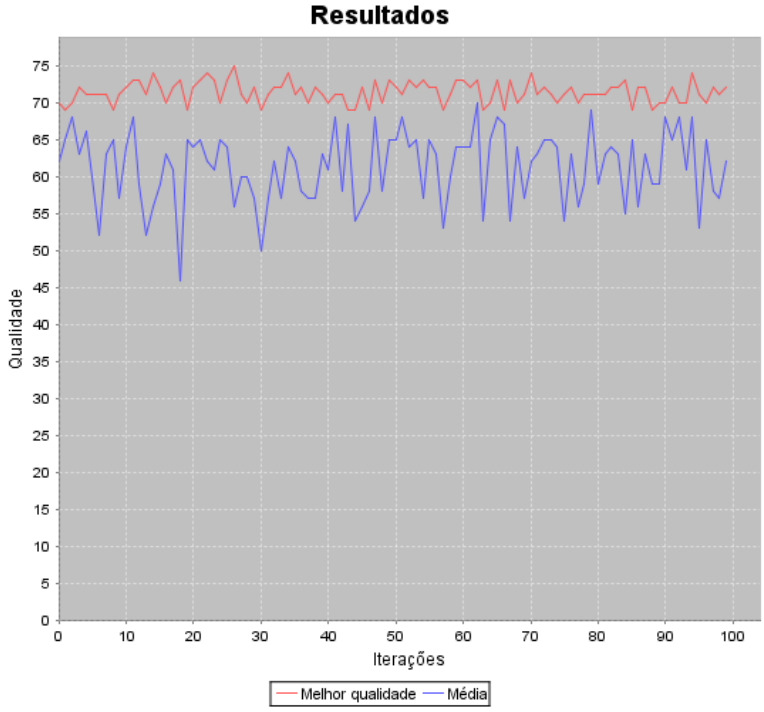

Figura 19 - Resultados obtidos da aplicação do EAS com heurística com 100 iterações

As figuras 18 e 19 mostram claramente que a heurística ajuda na obtenção de melhores resultados. Os valores da média da figura 18, praticamente variam de iteração para iteração de forma brusca, de valores muito elevados para muito mais baixos. A figura 19 não demonstra uma linha com valores constantes de iteração para

iteração como foi demonstrado no subcapítulo anterior, ver ponto 4.2.2, mas a variação da média não apresenta picos tão bruscos como na figura 18.

4.2.4 Resultados dos Testes aplicados à Sequência 4

Tabela 7 – Resultados dos testes para a sequência 4

	H	Nº it	%	Qualidade	Média	MS
AS	N	10	80	-108	-43	-111
		50	40	-107	-76	-114
		100	50	-109	-64	-114
	S	10	40	-103	-77	-107
		50	30	-103	-82	-112
		100	0	-105	-101	-109
EAS	N	10	80	-107	-57	-107
		50	90	-103	-44	-113
		100	70	-112	-51	-114
	S	10	40	-106	-83	-109
		50	30	-103	-83	-111
		100	40	-105	-73	-111
RAS	N	10	50	-104	-69	-108
		50	30	-106	-81	-112
		100	80	-104	-47	-113
	S	10	40	-104	-79	-107
		50	20	-107	-92	-110
		100	50	-104	-74	-110
Max-Min	N	10	60	-106	-70	-110
		50	70	-102	-58	-110

AS		100	40	-108	-76	-108
	S	10	30	-105	-80	-108
		50	30	-104	-84	-107
		100	30	-106	-84	-108

À semelhança dos resultados obtidos na tabela 6, os dados da tabela 7 revelam que os valores da qualidade e da melhor solução não variam muito de uns algoritmos para os outros. Os valores da média têm variações consoante a percentagem de inválidos, quanto maior essa percentagem, mais baixa é a média. Podemos verificar no resultado obtido no AS com 100 iterações e com a aplicação da heurística, que obteve uma percentagem de inválidos 0 e a média chegou aos -101. Já no resultado do EAS com 50 iterações e sem heurística, obteve 90% de inválidos e uma média de -44.

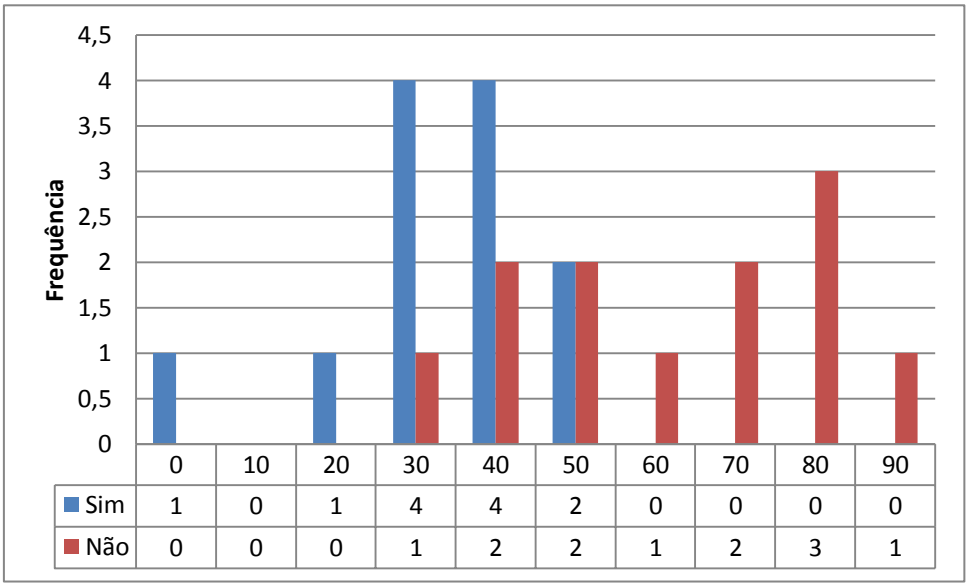

Figura 20 - Percentagem de inválidos na última iteração de experiências com e sem heurística

Os dados da figura 20 mostram que de certo modo a heurística ajudou na busca de melhores resultados, em que apenas a percentagem de inválidos só chegou aos 50%. Na maior parte das experiências sem heurística, a percentagem de inválidos está entre os 40% e os 90%.

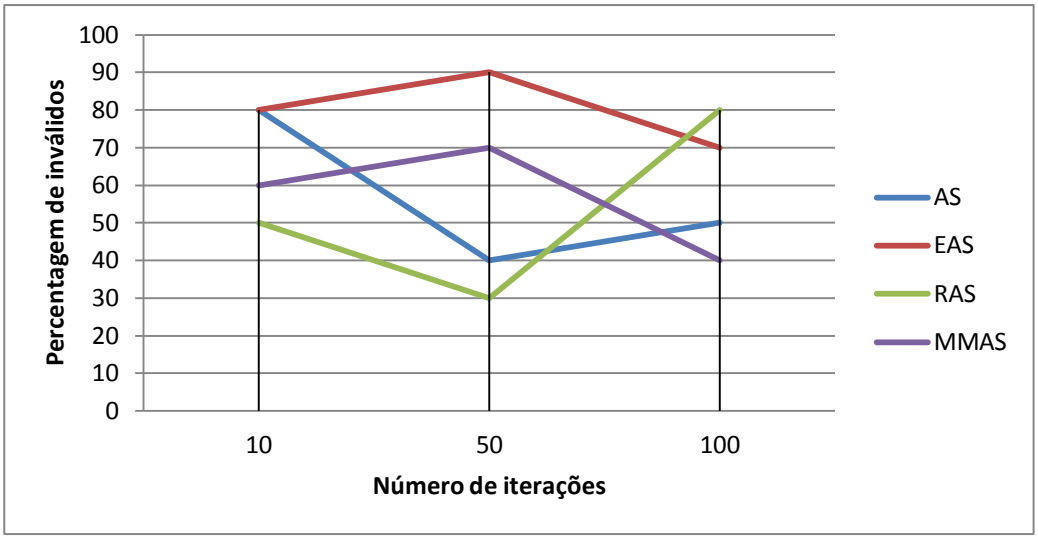

Figura 21 - Resultados obtidos da percentagem de inválidos em relação ao número de iterações, sem heurística

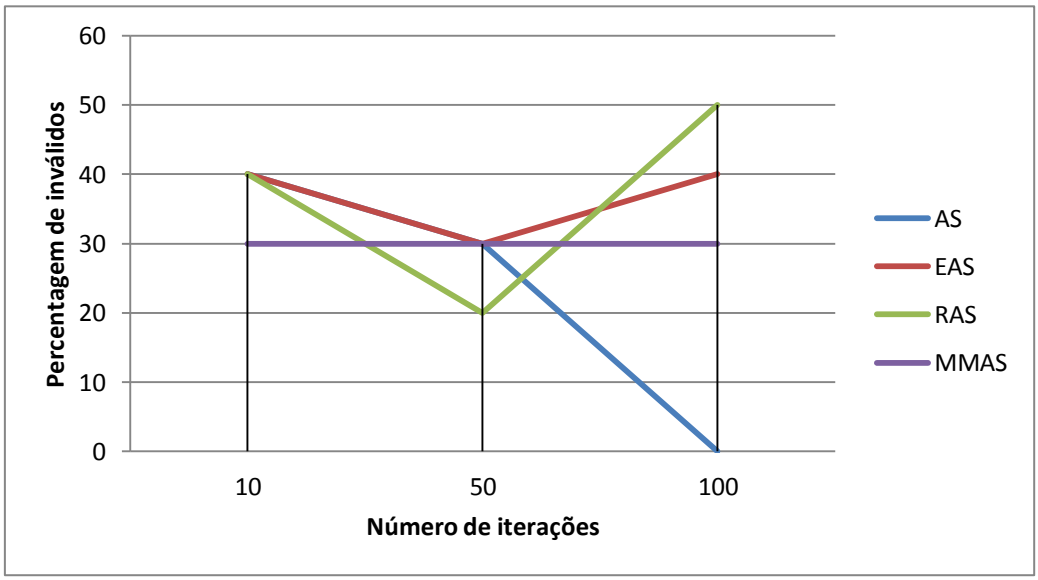

Figura 22 - Resultados obtidos da percentagem de inválidos em relação ao número de iterações, com heurística

Na figura 21, verifica-se que apenas dois algoritmos baixaram a percentagem de inválidos (EAS e MMAS), em relação à que obtiveram com 10 iterações. Na figura

22, o algoritmo AS baixou para 0 a percentagem de inválidos e o MMAS manteve o valor em todo o conjunto de iterações.

As conclusões que se podem tirar, de acordo com os resultados obtidos, são:

- A heurística desenvolvida, reduz claramente o número de soluções inválidas.
- A heurística ajuda a encontrar melhores soluções.
- Não há diferenças significativas no desempenho das quatro variantes que foram testadas.

4.2.5 Testes complementares

De modo a complementar os testes realizados nos pontos 4.2.1, 4.2.2, 4.2.3 e 4.2.4, foram realizados alguns testes adicionais para perceber o impacto dos seguintes parâmetros: número de formigas, ρ, α e β.

O método de optimização está parametrizado com as configurações indicadas na tabela 8.

Tabela 8 – Parâmetros dos testes complementares

Algoritmos	AS, EAS, RAS, MMAS
Sequência	4
Heurística (H)	Sim (S)
Número de Formigas	50
Número de iterações (Nº it)	10, 50, 100
α	1, 3
β	1, 3
ρ	0.1, 0.5
w	25
Número de repetições	1

Nas tabelas com os resultados é apresentada a seguinte informação: percentagem de inválidos (%), melhor solução da última iteração (Qualidade), média das soluções da última iteração (Média) e a melhor solução encontrada pelo algoritmo (MS).

Os parâmetros utilizados para obter os resultados da tabela 9 foram:

- $\rho = 0.5$
- $\alpha = 1$
- $\beta = 1$

Tabela 9 – Resultados dos testes com 50 formigas aplicados à sequência 4

	Nº it	**%**	**Qualidade**	**Média**	**MS**
AS	10	36	-106	-80	-110
	50	38	-105	-78	-111
	100	36	-107	-80	-112
EAS	10	30	-107	-81	-111
	50	22	-112	-89	-112
	100	34	-109	-81	-112
RAS	10	36	-106	-80	-109
	50	24	-109	-88	-109
	100	38	-107	-78	-113
Max-Min AS	10	26	-109	-85	-109
	50	22	-107	-87	-110
	100	32	-107	-80	-111

Tomando como ponto de partida os resultados obtidos anteriormente, ver ponto 4.2.4, podemos ver que o aumento do número de formigas ajuda na obtenção de melhores soluções, sendo por isso os valores da percentagem de inválidos mais baixos em todos os algoritmos.

Os parâmetros utilizados para obter os resultados da tabela 10 foram:

- $\rho = 0.1$
- $\alpha = 1$
- $\beta = 1$

Tabela 10 – Resultados dos testes com ρ a 0.1, aplicados à sequência 4

	Nº it	%	Qualidade	Média	MS
AS	10	36	-108	-83	-112
	50	36	-108	-81	-113
	100	32	-106	-83	-112
EAS	10	32	-109	-81	-109
	50	30	-107	-85	-112
	100	36	-108	-81	-111
RAS	10	32	-107	-82	-110
	50	39	-108	-80	-114
	100	36	-107	-80	-112
Max-Min AS	10	10	-106	-95	-108
	50	24	-106	-86	-115
	100	28	-106	-83	-110

Os resultados da tabela 10 indicam que a utilização de um valor mais baixo do ρ, ajudou na obtenção de melhores resultados do MMAS, visto que a percentagem de inválidos é mais baixa, em relação aos outros algoritmos.

Os parâmetros utilizados para obter os resultados da tabela 11 foram:

- $\rho = 0.5$
- $\alpha = 3$

- $\beta = 1$

Tabela 11 – Resultados dos testes com α a 3, aplicados à sequência 4

	Nº it	%	Qualidade	Média	MS
AS	10	36	-109	-78	-109
	50	30	-107	-83	-110
	100	32	-106	-83	-112
EAS	10	34	-107	-80	-108
	50	44	-106	-76	-111
	100	52	-107	-72	-112
RAS	10	38	-106	-77	-110
	50	36	-107	-78	-112
	100	30	-108	-85	-111
Max-Min AS	10	32	-106	-83	-109
	50	36	-107	-75	-109
	100	34	-106	-80	-111

Os resultados da tabela 11 são semelhantes em todos os algoritmos, apenas no EAS mostra uma subida na percentagem de inválidos.

Os parâmetros utilizados para obter os resultados da tabela 12 foram:

- $\rho = 0.5$
- $\alpha = 1$
- $\beta = 3$

Tabela 12 – Resultados dos testes com β a 3, aplicados à sequência 4

	Nº it	%	Qualidade	Média	MS
AS	10	34	-108	-82	-108
	50	18	-108	-92	-112
	100	26	-108	-87	-112
EAS	10	32	-107	-83	-109
	50	36	-106	-79	-110
	100	36	-107	-79	-112
RAS	10	28	-108	-85	-110
	50	32	-106	-83	-112
	100	36	-108	-80	-112

Os resultados da tabela 12 são semelhantes em todos os algoritmos, sendo que no MMAS os resultados foram inconclusivos.

4.3 Análise

As principais conclusões obtidas destes testes complementares são:

1. O aumento do número de formigas com o uso da heurística diminui o número de inválidos.
2. O ρ influencia os resultados do MMAS, quanto menor for, melhores resultados se obtém;
3. Não há diferenças significativas no desempenho das quatro variantes aplicando diferentes valores do α e β.

Uma questão que se levanta relativamente à conclusão 1 é se o facto da diminuição do número inválidos ser sinónimo de melhores resultados. Para analisar melhor esta questão, as figuras 23, 24, 25 e 26 mostram o cruzamento dos dados, melhor solução e média, das tabelas 7 e 9. As tabelas 7 e 9 apresentam os resultados obtidos com 10 e 50 formigas respectivamente e com heurística, que são os parâmetros que interessam para esta análise.

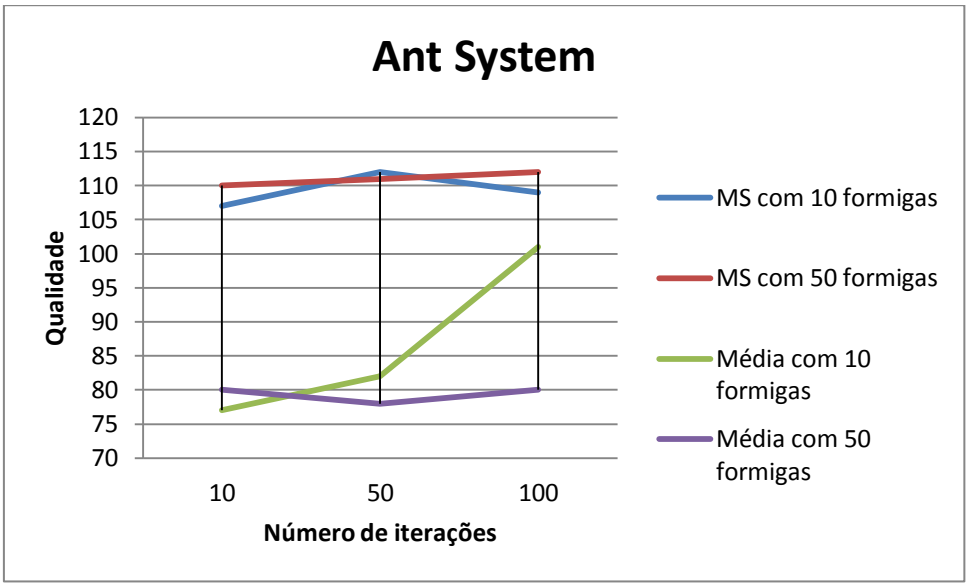

Figura 23 – Cruzamento dos dados obtidos relativamente ao Ant System

A figura 23 mostra os resultados obtidos relativamente ao Ant System e podemos verificar que:

- Duas das melhores soluções, com 10 e 100 iterações, são superiores o que indica a melhor solução de todas é superior com 50 formigas.
- A média das soluções é superior com 10 formigas.

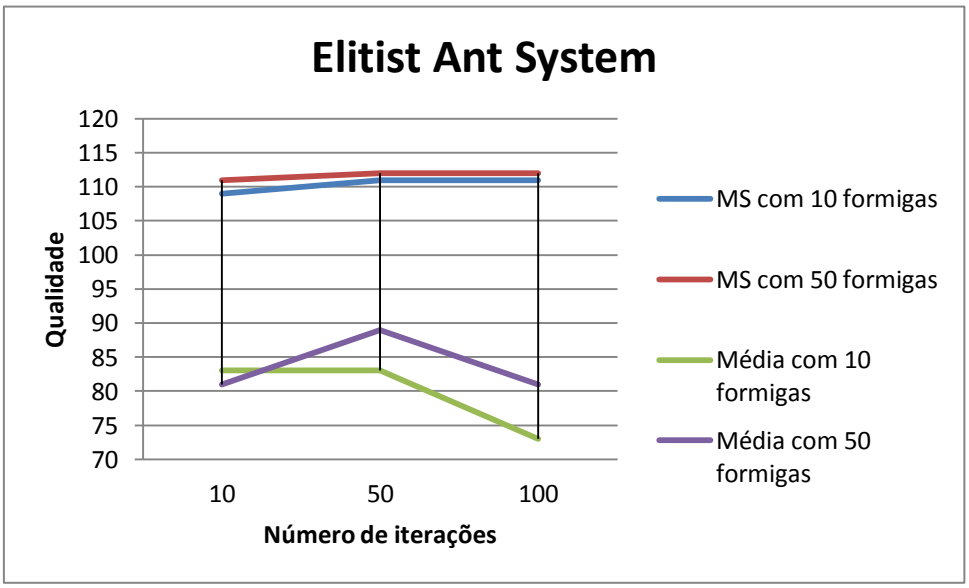

Figura 24 - Cruzamento dos dados obtidos relativamente ao Elitist Ant System

A figura 24 mostra os resultados obtidos relativamente ao Elitist Ant System e podemos verificar que:

- Todas as soluções obtidas são superiores com 50 formigas.
- A média das soluções é superior com 50 formigas, apesar do primeiro valor ser inferior.

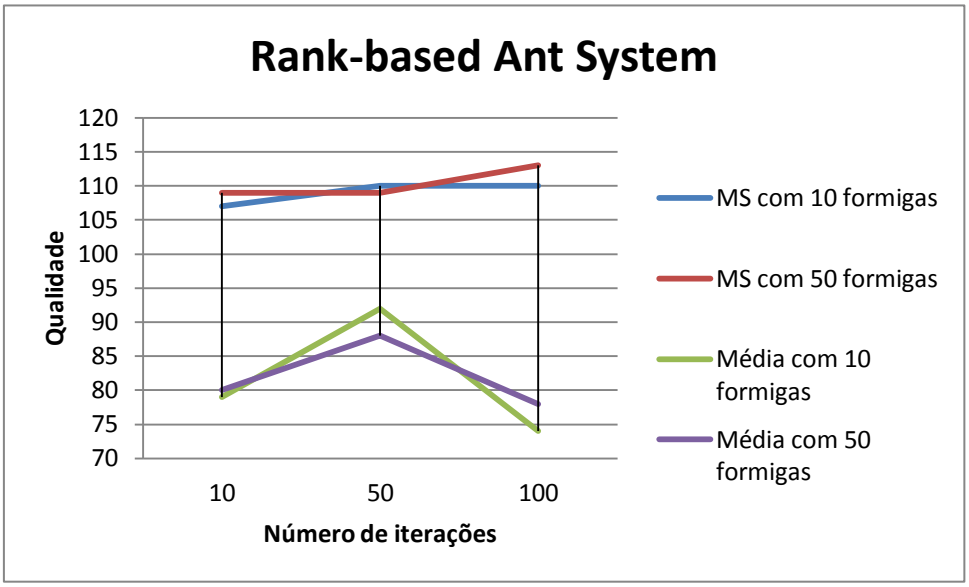

Figura 25 - Cruzamento dos dados obtidos relativamente ao Rank-based Ant System

A figura 25 mostra os resultados obtidos relativamente ao Rank-based Ant System e podemos verificar que:

- A melhor solução de todas é superior com 50 formigas, com 10 e 100 iterações.
- A média das soluções é superior com 50 formigas, com 10 e 100 iterações.

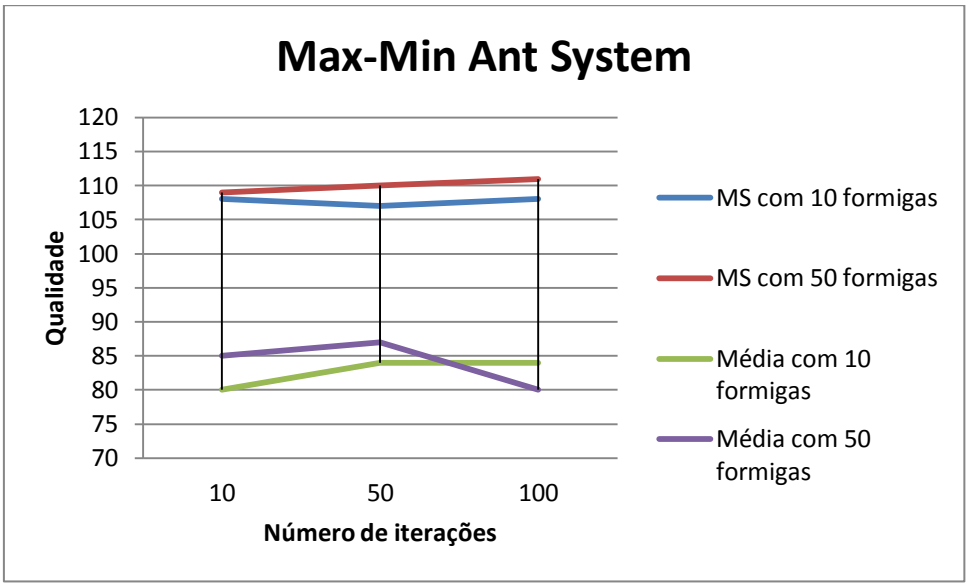

Figura 26- Cruzamento dos dados obtidos relativamente ao Max-Min Ant System

A figura 26 mostra os resultados obtidos relativamente ao Max-Min Ant System e podemos verificar que:

- Todas as soluções obtidas são superiores com 50 formigas.
- A média das soluções é superior com 50 formigas, com 10 e 50 iterações.

Com a análise destes dados, das figuras 23, 24, 25 e 26, podemos concluir que com a heurística e com o aumento do número de formigas, tendo em conta que o tempo de computação também aumentou, para além da diminuição do número de inválidos, obtemos melhores soluções, nas quatro variantes implementadas.

5 Aplicação

5.1 Descrição Geral

Uma das principais componentes do projecto consiste no desenvolvimento de uma aplicação web desenvolvida na tecnologia JavaServer Pages (JSP) que pretenda resolver problemas HP segundo um conjunto de parâmetros definidos à priori, de modo a que qualquer utilizador tenha a possibilidade de ver a resolução de um qualquer problema. O utilizador pode utilizar a configuração por defeito ou também inserir os próprios dados de acordo com os campos que o sistema apresenta. O ambiente de desenvolvimento que serviu para a construção deste projecto foi constituído por:

- Máquina (Computador Portátil) de desenvolvimento que incluía a máquina virtual assim como todas as ferramentas de apoio como NetBeans IDE, GlassFish Server, utilizadas neste projecto.

A tecnologia JSP foi escolhida pelo facto de ter sido leccionada neste mestrado, e sendo por isso uma tecnologia à qual me sentia mais à vontade de trabalhar. Quando esta tecnologia foi leccionada, foram utilizadas as ferramentas NetBeans IDE e GlassFish Server, e decidi utilizá-las neste projecto.

Esta aplicação tem como principais objectivos:

- Dar a conhecer os temas abordados neste projecto, como ACO, modelos HP e os algoritmos implementados baseados em colónias de formigas.
- Servir como ferramenta para resolver problemas de previsão de estrutura de proteínas, onde são aplicados os algoritmos implementados.

5.2 Funcionalidades

A recolha das funcionalidades da aplicação desenvolvida, foi feita de acordo com os parâmetros que são necessários para a resolução de problemas HP. De seguida é apresentada uma listagem das funcionalidades da aplicação desenvolvida neste projeto:

1. **Resolver problema HP**

Esta funcionalidade permite ao utilizador resolver problemas HP. Para isto, o sistema disponibiliza um conjunto de parâmetros definidos por omissão, permitindo ao utilizador resolver problemas sem a inserção de qualquer dado. Por outro lado, o utilizador poderá introduzir os seus próprios parâmetros de acordo com os campos disponibilizados pelo sistema. De seguida serão apresentados os campos necessários para a resolução de problemas HP.

1.1. Input Problema

1.1.1. Inserir sequência. A sequência, que é o problema que se quer resolver, com um máximo de 100 caracteres, contém as seguintes opções:

i. **Sequência aleatória**, opção seleccionada por omissão, onde o sistema escolhe uma sequência aleatória para ser resolvida.

ii. **Inserir sequência**, onde o utilizador introduz uma sequência nova à sua escolha.

iii. **Fazer upload de ficheiro**, onde o utilizador pode fazer o upload de um ficheiro com o formato text file (*.txt*), que contém a sequência.

1.2 Escolha e parametrização das variáveis

1.2.1 Algoritmo, que envolve o tipo e heurística.

i. O tipo de Algoritmo indica a vertente que vai ser aplicada na resolução do problema, que o utilizador pode escolher Ant System, Elitist Ant System, Rank-based Ant System e Max-Min Ant system.

ii. A heurística está envolvida directamente com o algoritmo escolhido, mas também com os parâmetros alfa e beta, descritos no próximo tópico.

1.2.2 Parâmetros

i. **Número de formigas**, com um número mínimo de 5 e máximo de 100 formigas que são responsáveis pela construção das soluções.

ii. **Número de iterações**, com um número mínimo de 10 e máximo de 1000 iterações, indica quantas as vezes, o número de formigas escolhido anteriormente, vão construir soluções.

iii. **Rho**, com um número mínimo de 0 e máximo de 0.5.

iv. **Alfa**, com um número mínimo de 1 e máximo 3.

v. **Beta**, com um número mínimo de 1 e máximo 3.

1.3 Execução e apresentação de resultados

Esta funcionalidade permite que o utilizador execute e visualize os resultados do problema. Após a resolução de um problema, um conjunto de informações são apresentadas:

i. A melhor solução encontrada.

ii. Visualização detalhada da matriz de feromonas.

iii. Visualização detalha de soluções. Esta funcionalidade permite ao utilizador visualizar a melhor solução e a matriz de feromonas de cada iteração, através dos botões "anterior" e "seguinte".

iv. Exportar ficheiro. Esta funcionalidade permite ao utilizador exportar um ficheiro pdf com os principais resultados obtidos da resolução do problema. Este ficheiro contém os parâmetros introduzidos para resolver o problema, a melhor solução, a matriz de feromonas e o gráfico com os resultados que contêm a evolução da média das qualidades das soluções e da melhor solução.

2. Visualizar páginas de ajuda.

Esta funcionalidade permite ao utilizador obter um pouco de conhecimento sobre ACO, modelos HP e os algoritmos implementados, o Ant System, o Elitist Ant System, o Rank-based Ant System e o Max-Min Ant System.

5.3 Interfaces da aplicação

As interfaces da aplicação foram criadas utilizando a ferramenta NetBeans. Neste tópico é importante descrever cada interface realizada e ligar os interfaces às funcionalidades, visto que nem todas as interfaces dizem respeito a funcionalidades. Para a criação destas interfaces foi necessário a criação de páginas JSP, onde foi utilizado um template HTML, que contém todos os elementos constituintes de cada interface. As interfaces que pertencem a esta aplicação são: (1) Resolver Problema, (2) Início, (3) ACO, (4) Modelos HP, (5) Ant System, (6) EAS, RAS e MMAS e (7) Sobre.

5.3.1 Resolver Problema

Esta interface corresponde às funcionalidades resolver problema, visualizar resultados, visualizar soluções, exportar ficheiro e requer a introdução de dados do utilizador. A página JSP (Problema.jsp), demonstrada na figura 27, começa por apresentar um conjunto de campos para o utilizador introduzir dados, que corresponde à funcionalidade resolver problema.

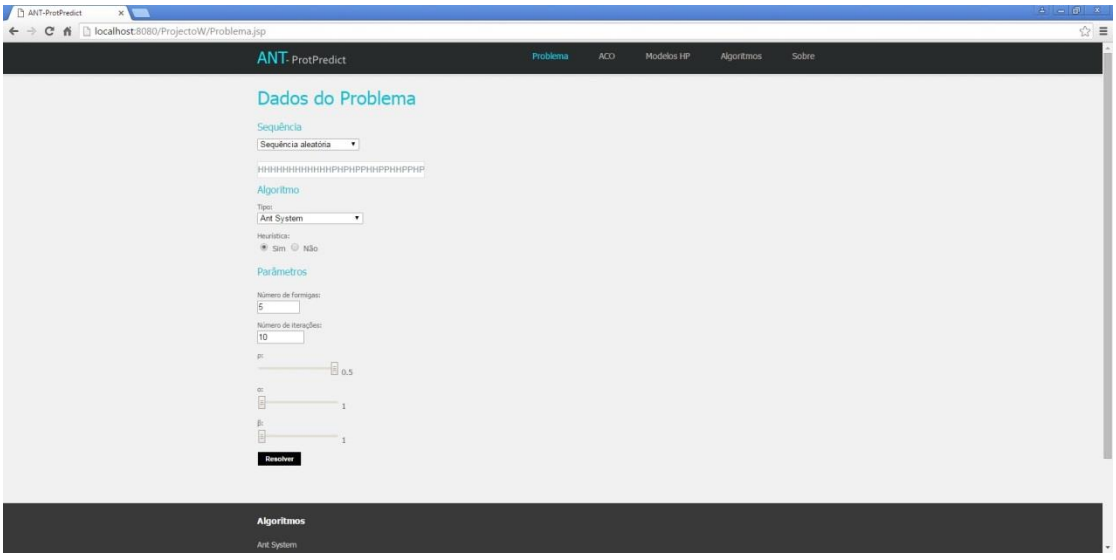

Figura 27 – Interface gráfica da página Problema.jsp

De seguida, vai ser descrito como os campos, descritos no ponto 5.2, foram implementados:

1.1 Input Problema

1.1.1 Inserir sequência. Este campo, que contém as opções: sequência aleatória, inserir sequência e fazer upload de ficheiro, e apenas uma pode ser seleccionada, foi utilizado um *select* com as opções referidas, como é mostrado na figura 28.

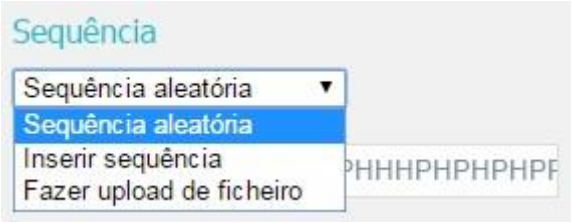

Figura 28 – Opções para inserir sequência

Quando o utilizador muda a opção, é chamada uma função em *javascript*, que o objectivo é de acordo com o seguinte:

i. **Sequência aleatória**, é mostrado um *input* do tipo *text*, com uma sequência introduzida, que foi gerada por outra função, que tem o objectivo de gerar um número e de acordo com esse número, devolver a sequência correspondente.

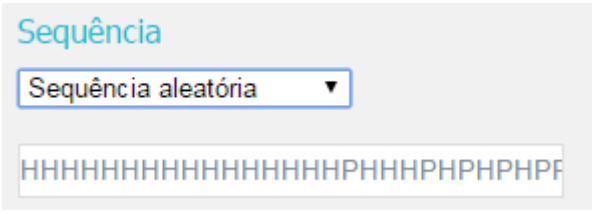

Figura 29 – Opção "Sequência aleatória" seleccionada

ii. **Inserir sequência**, é mostrado um *input* do tipo *text* vazia, para o utilizador introduzir uma sequência.

Figura 30 – Opção "Inserir Sequência" seleccionada

iii. **Fazer upload de ficheiro**, é mostrado um *input* do tipo *file*, que permite ao utilizador escolher o ficheiro com a sequência.

Existem duas *form* nesta página JSP para esta funcionalidade, para as opções i e ii é utilizada a *form* com *method get*, e para a opção iii é utilizada a *form* com *method post.*

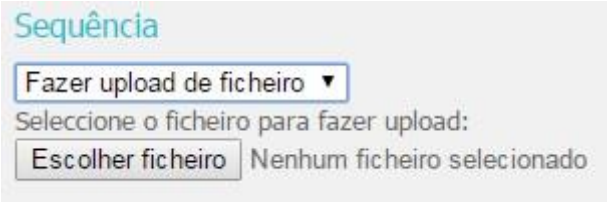

Figura 31 - Opção "Fazer upload de ficheiro" seleccionada

1.2 Escolha e parametrização das variáveis

1.2.1 Algoritmo

Figura 32 – Parâmetros do Algoritmo

i. No campo tipo de Algoritmo, demonstrado na figura 33, foi utilizado um *select* com as opções Ant System, Elitist Ant System, Rank-based Ant System e Max-Min Ant system. A opção Ant System está seleccionada inicialmente, por omissão.

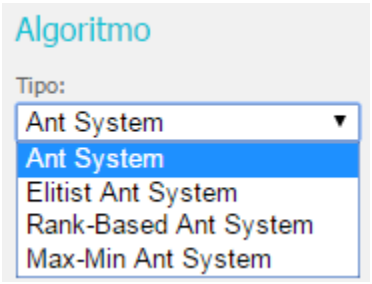

Figura 33 – Opções do tipo de algoritmo

ii. No campo heurística, demonstrado na figura 31, foi utilizado dois *radio button*, com as opções sim e não, para o utilizador seleccionar apenas uma delas. A opção sim está seleccionada inicialmente, por omissão.

1.2.2 Parâmetros

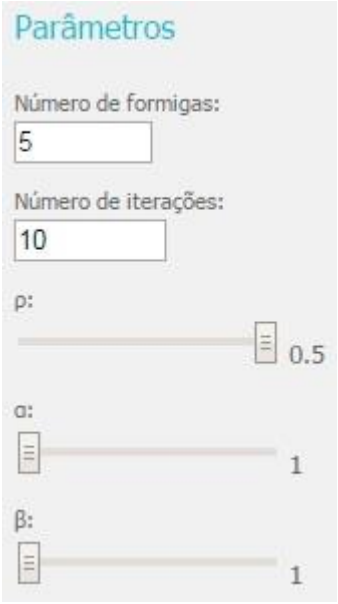

Figura 34 - Parâmetros

i. **Número de formigas**, foi utilizado um campo do tipo *input* do tipo *number* com o valor mínimo 5 e máximo 100. O valor 5 está seleccionado inicialmente, por omissão.

ii. **Número de iterações**, foi utilizado um campo do tipo *input* do tipo *number* com o valor mínimo 10 e máximo 1000. O valor 10 está seleccionado inicialmente, por omissão.

iii. **Rho,** foi utilizado um campo do tipo *input* do tipo *range* com o valor mínimo 0 e máximo 0.5. O valor 0.5 está seleccionado inicialmente, por omissão.

iv. **Alfa**, foi utilizado um campo do tipo *input* do tipo *range* com o valor mínimo 1 e máximo 3. O valor 1 está seleccionado inicialmente, por omissão.

v. **Beta**, foi utilizado um campo do tipo *input* do tipo *range* com o valor mínimo 1 e máximo 3. O valor 1 está seleccionado inicialmente, por omissão.

Após a introdução de novos dados, ou o utilizador optar usar os dados pré-definidos e clicar no botão *Resolver,* os dados são enviados para a *servlet* e esta aplica a resolução do problema de acordo com os dados introduzidos como é demonstrado na figura 35.

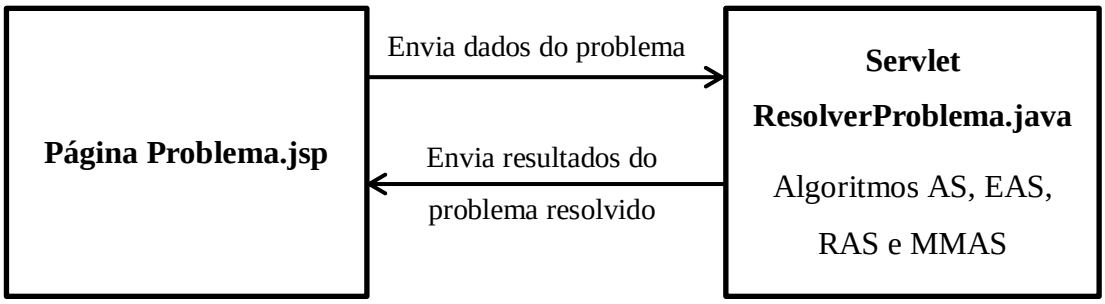

Figura 35 – Esquema de funcionamento da resolução de um problema

A Servlet ResolverProblema.java contém a implementação dos algoritmos AS, EAS, RAS e MMAS, e de acordo com os dados recebidos que inclusivamente indicam qual o algoritmo a aplicar, resolve o problema pretendido. Após a resolução do problema com o algoritmo indicado e os dados necessários, são guardados os resultados num objecto e através de um *request.setAttribute*, a página Problema.jsp vai ter acesso a esses dados através de *request.getAttribute* e mostrá-los.

A página JSP apresenta um conjunto de informações, que correspondem à funcionalidade visualizar resultados, que são:

a. A melhor solução (figura 36), onde são apresentados dados sobre esta como: a sequência que foi utilizada, a iteração, a percentagem de inválidos nessa iteração, a média das qualidades na iteração, os valores mínimo e máximo (no caso de ter escolhido o algoritmo Max-Min AS). Este conjunto de dados é apresentado através da componente *label*. A grelha de duas dimensões é apresentada com o uso da componente *table* e código java para mostrar a informação que se pretende. Para a grelha ser apresentada de forma a apenas focar a solução, e descartar células vazias, são efectuados cálculos em código java, para esse efeito.

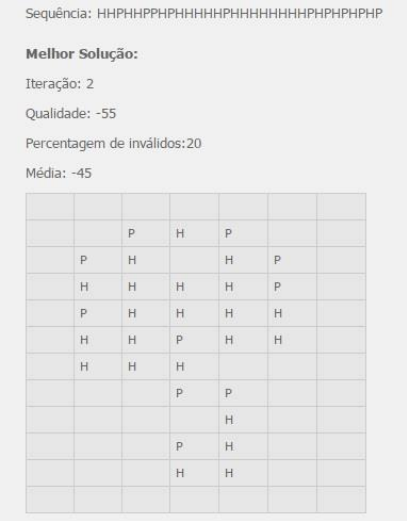

Figura 36 – Matriz da melhor solução

b. Visualização detalhada da matriz de feromonas, demonstrada na figura 37. À semelhança da apresentação da melhor solução. A grelha de duas dimensões é apresentada com o uso da componente *table* e código java para mostrar a informação que se pretende. Para a grelha ser apresentada de forma a apenas focar a solução, e descartar células vazias, são efectuados cálculos em código java, para esse efeito. Esta grelha mostra as células com cores, vários tipos de tons de cinzento. Cada tom representa a percentagem de "passagens" naquela célula pelas formigas, ou seja, o tom mais claro indica que houve poucas "passagens" naquela célula, enquanto que o mais escuro indica que houve muitas "passagens" e assim sucessivamente.

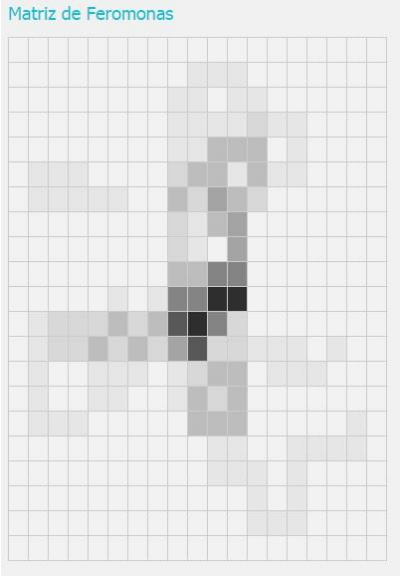

Figura 37 – Matriz de Feromonas

c. Visualização detalhada de soluções. A funcionalidade "visualização detalhada de soluções" como é mostrada na figura 38, foi implementada utilizando dois botões "anterior" e "seguinte". Para o sistema saber se os botões foram clicados, foram criadas duas funções em *javascript*, uma para cada botão. Estas funções imprimem na *div* chamada *novadiv*, o que está na página Resultado.jsp. Como o objectivo destes botões, é mostrar a melhor solução e matriz de feromonas da iteração anterior ou seguinte, consoante o botão clicado, é necessário saber na página Resultado.jsp, qual o botão clicado e para isso é enviado um parâmetro a indicar isso mesmo. Na página Resultado.jsp, apenas é calculada a posição de acordo com o parâmetro escolhido, e colocar em duas *table* a solução e a matriz de feromonas, sendo esta depois mostrada na *div*, na página Problema.jsp.

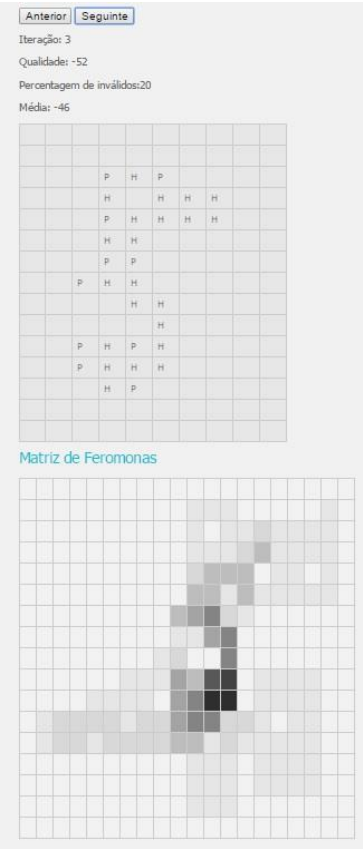

Figura 38 – Visualizar soluções

d. Exportar ficheiro. No fundo da página (Problema.jsp), depois da apresentação dos resultados, é mostrado um botão "Exportar para PDF", que diz respeito à funcionalidade "Exportar ficheiro". De acordo com os objectivos desta funcionalidade, ao clicar no botão aparece de seguida o pdf com os resultados. A implementação passou por criar uma página JSP (exportar.jsp), e recorrendo à biblioteca *itextpdf* e aos dados necessários para a construção do documento, foi criado o documento pdf. Na página Problema.jsp foi guardado um objecto na sessão com a informação necessária para a construção do documento pdf, onde foi acedida na página exportar.jsp. A elaboração do documento foi feita através da criação de um objecto Document, e foram adicionadas as informações, uma a uma, desde informações sobre os dados introduzidos para resolver o problema, como as matrizes da melhor solução e feromonas, e até o gráfico. A figura 39 mostra o pdf gerado.

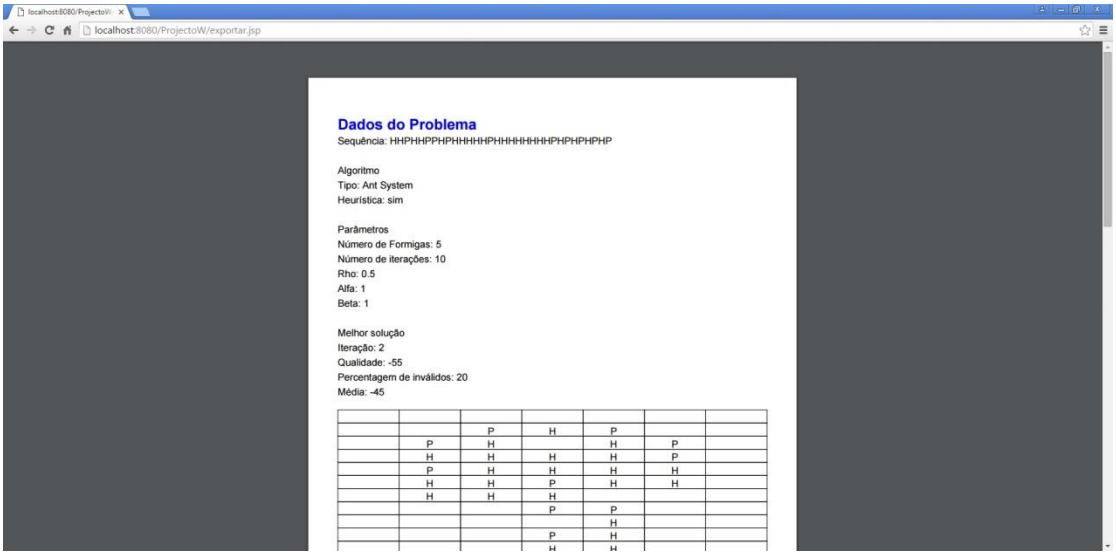

Figura 39 – Pdf gerado pela aplicação

O gráfico gerado pela aplicação e introduzido no pdf, indica como se comportou o algoritmo na construção das soluções. Para a construção deste gráfico, foi utilizado o *JFreeChart* para criar um gráfico do tipo *XYLineChart*. A concepção do gráfico, envolveu a criação de dois XYSeries, onde são adicionados os dados a serem utilizados e a criação de um XYSeriesCollection para adicionar os XYSeries criados anteriormente. De seguida faz-se a criação do gráfico através de *XYLineChart* e para finalizar cria-se um XYPlot para definir os parâmetros do gráfico, como a cor de fundo. Este gráfico foi guardado como imagem no formato *.PNG*, e para isso foi

necessário criar um *file* a indicar o caminho da pasta a ser guardado, já com o nome completo do ficheiro, e utilizando o *ChartUtilities*, o gráfico é guardado no ficheiro criado anteriormente. A figura 40 mostra o aspecto do gráfico gerado pela aplicação.

Figura 40 – Gráfico gerado pela aplicação

5.2.1 Páginas adicionais

As páginas Início.jsp, ACO.jsp, ModelosHP.jsp, AntSystem.jsp, Algoritmos.jsp, e Sobre.jsp, contêm uma descrição do problema de previsão das estruturas de proteínas assim como a descrição de cada algoritmo implementado na aplicação. As figuras seguintes demonstram a interface gráfica dessas páginas.

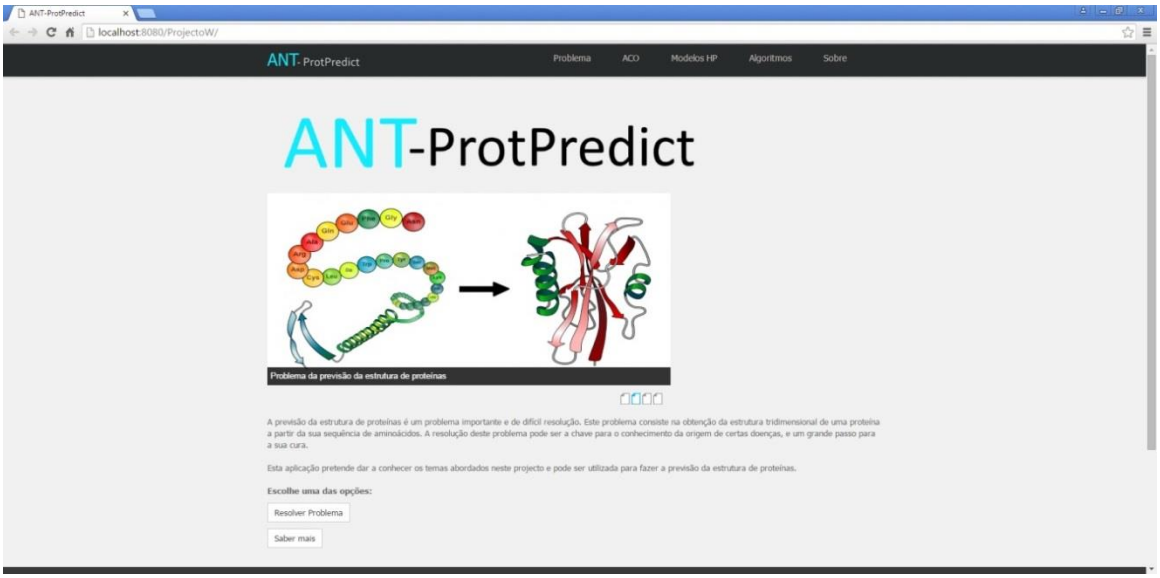

Figura 41 – Interface gráfica da página Início.jsp

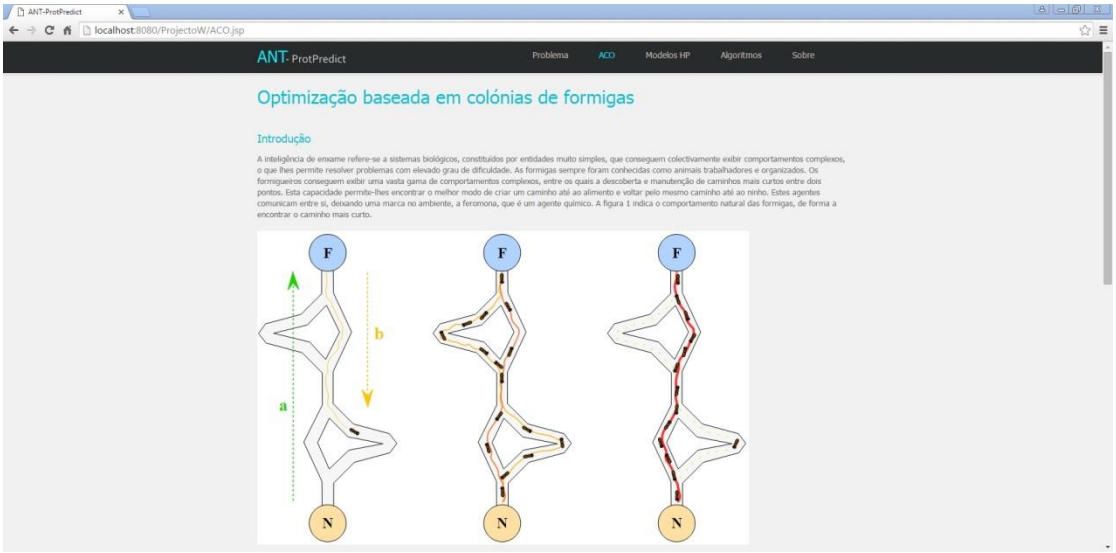

Figura 42 – Interface gráfica da página ACO.jsp

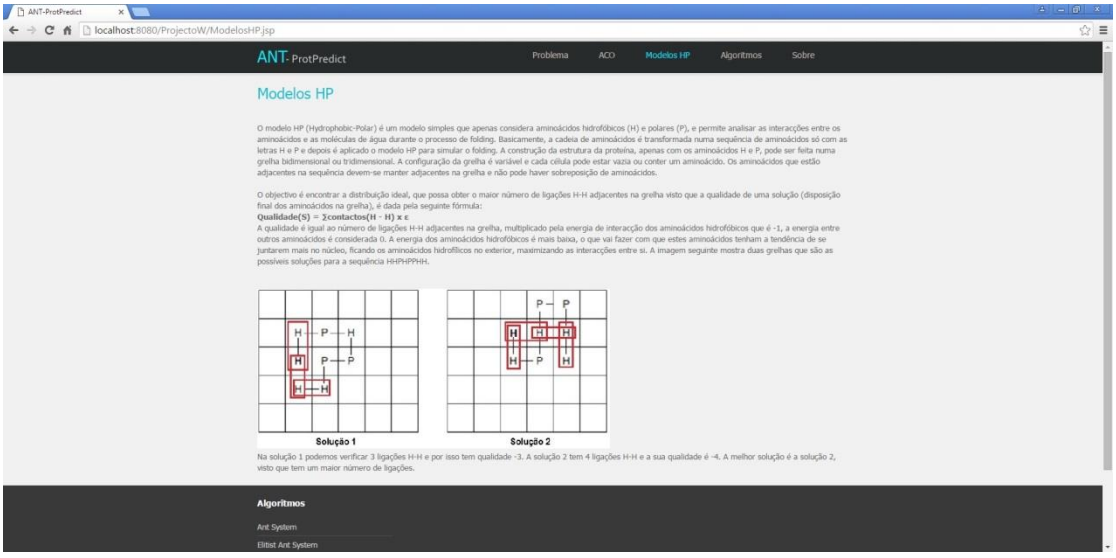

Figura 43 – Interface gráfica da página ModelosHP.jsp

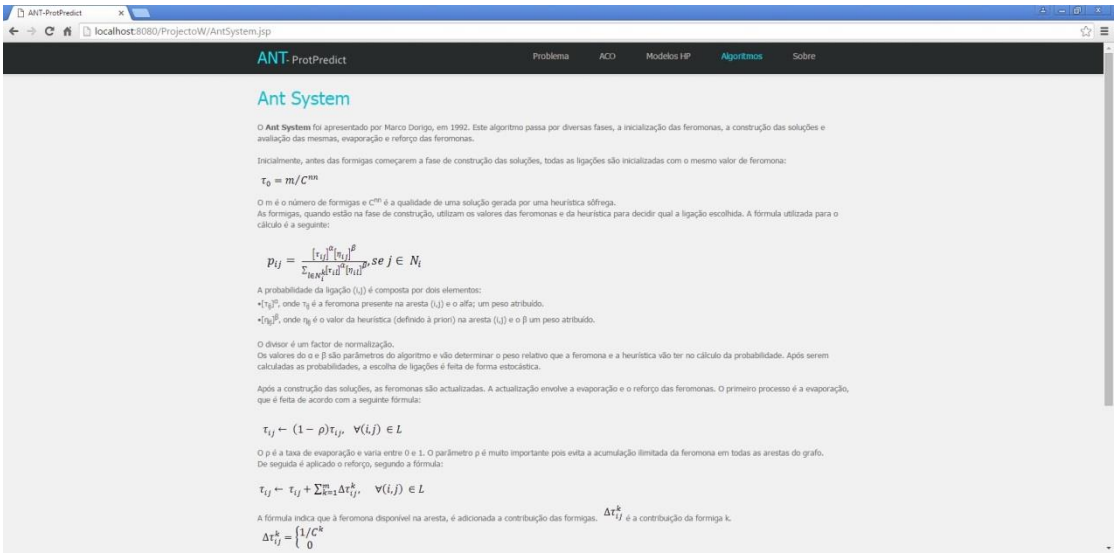

Figura 44 – Interface gráfica da página AntSystem.jsp

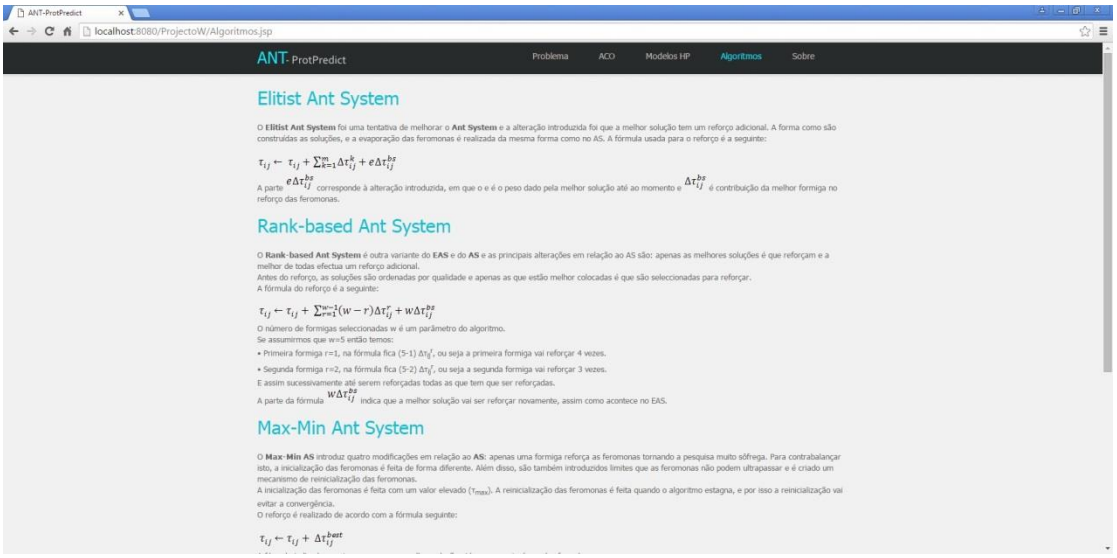

Figura 45 – Interface gráfica da página Algoritmos.jsp

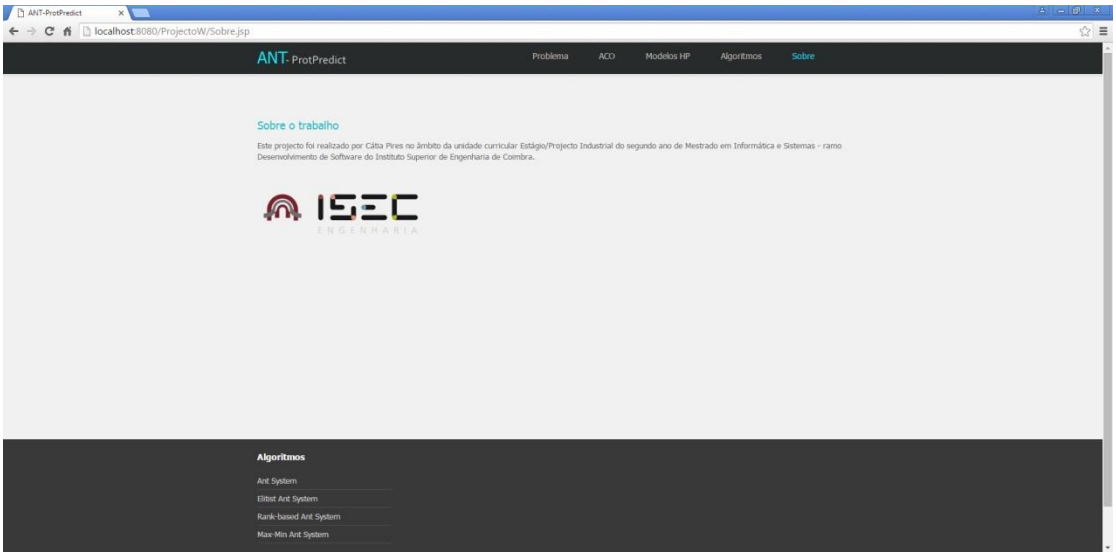

Figura 46 - Interface gráfica da página Sobre.jsp

6 Conclusão

6.1 Resultados do Projecto

Este projecto tinha um duplo objectivo: conhecer, desenvolver e testar algoritmos baseados em colónias de formigas para resolver problemas de previsão com modelos HP e a implementação de uma aplicação computacional baseada em ACO, para prever a estrutura de proteínas, onde são aplicados os algoritmos Ant System, Elitist Ant System, Rank-based Ant System e Max-Min Ant System implementados.

No final do projecto é possível listar as principais contribuições:

- Implementação completa dos vários algoritmos baseados em ACO para optimização de modelos HP.
- Conjunto alargado de resultados de testes efectuados na optimização de modelos HP.
- Estes resultados permitiram obter um conjunto de conclusões relevantes:
    - A heurística desenvolvida, reduz claramente o número de soluções inválidas.
    - O aumento do número de formigas com o uso da heurística diminui o número de inválidos, e ajuda na obtenção de melhores soluções.
    - Não há diferenças significativas no desempenho das quatro variantes que foram testadas.
    - O ρ influencia os resultados do MMAS, quanto menor for, melhores resultados se obtém.
    - Não há diferenças significativas no desempenho das quatro variantes aplicando diferentes valores do α e β.
- A aplicação web funcional, desenvolvida na tecnologia JSP, que baseia-se em ACO (Optimização baseada em colónias de formigas) para prever a estrutura de proteínas, onde são aplicados os algoritmos implementados. A aplicação destina-se a quem queira aprofundar os conhecimentos sobre ACO's e modelos HP. Por outro lado, pode também ser utilizada como ferramenta de

trabalho ou de investigação, por pessoas que tenham a intenção de realizar os próprios testes.

6.2 Trabalho Futuro

Os algoritmos e a aplicação web foram realizados de acordo com o que foi proposto mas evidentemente este trabalho pode ser expandido no futuro. Sendo esta aplicação web, pioneira no ambiente virtual, seria importante expandi-la.

As tarefas que podem ser melhoradas na aplicação web são:

- Implementação de mais algoritmos;
- Tornar o aspecto gráfico mais interactivo, ou seja, o utilizador ter a oportunidade de visualizar a formiga a construir a solução passo a passo;
- Ter a opção dos algoritmos serem aplicados num ambiente 3D;
- Criação de utilizadores, de modo a que os utilizadores possam visualizar o histórico dos problemas resolvidos.

6.3 Apreciação Crítica do Estágio

A nível pessoal, este projecto, foi uma experiência gratificante pelo facto de ser uma experiência à qual procurava, a investigação. Uma investigação mais aprofundada e exigente do que um mero trabalho académico possa exigir.

O desenvolvimento de um projecto, individualmente, é sem dúvida um grande desafio e visa aplicar os conhecimentos adquiridos e reactivar a capacidade de procurar novos conhecimentos, para novos domínios.

Neste sentido, este projecto contribuiu para aprofundar os meus conhecimentos na área de bioinformática, relativamente a algoritmos baseados em colónias formigas, modelos HP e principalmente perceber o problema da previsão de estrutura de proteínas. A resolução de problemas da previsão de estruturas combinando ACO com modelos HP, aplicar algoritmos e perceber os resultados obtidos através de testes, foi desafiante. A criação da aplicação em JSP permitiu relembrar e aplicar os

conhecimentos adquiridos na disciplina de Plataformas de Desenvolvimento deste Mestrado.

A contribuição com o desenvolvimento de uma aplicação, reflexo de todos os conhecimentos adquiridos e a finalização deste projecto, é uma razão de orgulho e satisfação.

Referências

[1] M. Dorigo, V. Maniezzo, and A. Colorni, "Positive feedback as a search strategy," Dipartimento di Elettronica, Politecnico di Milano, Italy, Tech. Rep. 91-016, 1991.

[2] M. Dorigo, V. Maniezzo, and A. Colorni, "Ant System: Optimization by a colony of cooperating agents," *IEEE Transactions on Systems, Man, and Cybernetics – Part B*, vol. 26, no. 1, pp. 29–41, 1996.

[3] ——, "Ant Colony System: A cooperative learning approach to the traveling salesman problem," *IEEE Transactions on Evolutionary Computation*, vol. 1, no. 1, pp. 53–66, 1997.

[4] T. Stützle and H. H. Hoos, "MAX–MIN Ant System," *Future Generation Computer Systems*, vol. 16, no. 8, pp. 889–914, 2000.

[5] T. St¨utzle and H. H. Hoos, "The MAX–MIN Ant System and local search for the traveling salesman problem," in *Proceedings of the 1997 IEEE International Conference on Evolutionary Computation (ICEC'97)*, T. B¨ack *et al.*, Eds. IEEE Press, Piscataway, NJ, 1997, pp. 309–314.

[6] L. M. Gambardella, E. D. Taillard, and G. Agazzi, "MACS-VRPTW: A multiple ant colony system for vehicle routing problems with time windows," in *New Ideas in Optimization*, D. Corne *et al.*, Eds. McGraw Hill, London, UK, 1999, pp. 63–76.

[7] M. Reimann, K. Doerner, and R. F. Hartl, "D-ants: Savings based ants divide and conquer the vehicle routing problems," *Computers & Operations Research*, vol. 31, no. 4, pp. 563–591, 2004.

[8] L. M. Gambardella and M. Dorigo, "Ant Colony System hybridized with a new local search for the sequential ordering problem," *INFORMS Journal on Computing*, vol. 12, no. 3, pp. 237–255, 2000.

[9] V. Maniezzo, "Exact and approximate nondeterministic tree-search procedures for the quadratic assignment problem," *INFORMS Journal on Computing*, vol. 11, no. 4, pp. 358–369, 1999.

[10] K. Socha, J. Knowles, and M. Sampels, "AMAX-MIN ant system for the university timetabling problem," in *Ant Algorithms, 3rd International Workshop, ANTS 2002*, ser. LNCS, M. Dorigo *et al.*, Eds., vol. 2463. Berlin, Germany: Springer Verlag, 2002, p. 1.

[11] K. Socha, M. Sampels, and M. Manfrin, "Ant algorithms for the university course timetabling problem with regard to the state-ofthe- art," in *Applications of Evolutionary Computing, Proceedings of EvoWorkshops 2003*, ser. LNCS, G. R. Raidl *et al.*, Eds., vol. 2611. Springer Verlag, 2003, pp. 334–345

[12] D. Costa and A. Hertz, "Ants can colour graphs," *Journal of the Operational Research Society*, vol. 48, pp. 295–305, 1997.

[13] D. Merkle, M. Middendorf, and H. Schmeck, "Ant colony optimization for resource-constrained project scheduling," *IEEE Transactions on Evolutionary Computation*, vol. 6, no. 4, pp. 333–346, 2002.

[14] M. L. den Besten, T. St¨utzle, and M. Dorigo, "Ant colony optimization for the total weighted tardiness problem," in *Proceedings of PPSN-VI*, ser. LNCS, M. Schoenauer *et al.*, Eds., vol. 1917. Springer Verlag, 2000, pp. 611–620.

[15] D. Merkle and M. Middendorf, "Ant colony optimization with global pheromone evaluation for scheduling a single machine," *Applied Intelligence*, vol. 18, no. 1, pp. 105–111, 2003.

[16] C. Blum, "Beam-ACO—Hybridizing ant colony optimization with beam search: An application to open shop scheduling," *Computers & Operations Research*, vol. 32, no. 6, pp. 1565–1591, 2005.

[17] L. Lessing, I. Dumitrescu, and T. St¨utzle, "A comparison between ACO algorithms for the set covering problem," in *ANTS'2004, Fourth Internatinal Workshop on Ant Algorithms and Swarm Intelligence*, ser. LNCS, M. Dorigo *et al.*, Eds., vol. 3172. Springer Verlag, 2004, pp.1–12.

[18] C. Blum and M. J. Blesa, "New metaheuristic approaches for the edge-weighted k-cardinality tree problem," *Computers & Operations Research*, vol. 32, no. 6, pp. 1355–1377, 2005.

[19] G. Leguizam´on and Z. Michalewicz, "A new version of Ant System for subset problems," in *Proceedings of CEC'99*. IEEE Press, Piscataway, NJ, 1999, pp. 1459–1464.

[20] S. Fenet and C. Solnon, "Searching for maximum cliques with ant colony optimization," in *Applications of Evolutionary Computing, Proceedings of EvoWorkshops 2003*, ser. LNCS, G. R. Raidl *et al.*, Eds., vol. 2611. Springer Verlag, 2003, pp. 236–245.

[21] C. Solnon, "Solving permutation constraint satisfaction problems with arti cial ants," in *Proceedings of ECAI'2000*. Amsterdam, The Netherlands: IOS Press, 2000, pp. 118–122.
[22] ——, "Ants can solve constraint satisfaction problems," *IEEE Transactions on Evolutionary Computation*, vol. 6, no. 4, pp. 347–357, 2002.
[23] R. S. Parpinelli, H. S. Lopes, and A. A. Freitas, "Data mining with na ant colony optimization algorithm," *IEEE Transactions on Evolutionary Computation*, vol. 6, no. 4, pp. 321–332, 2002.
[24] D. Martens, M. D. Backer, R. Haesen, B. Baesens, C. Mues, and J. Vanthienen, "Ant-based approach to the knowledge fusion problem," in *Ant Colony Optimization and Swarm Intelligence, 5th International Workshop, ANTS 2006*, ser. LNCS, M. Dorigo *et al.*, Eds., vol. 4150. Springer Verlag, 2006, pp. 84–95.
[25] L. M. de Campos, J. M. Fern´andez-Luna, J. A. G´amez, and J. M. Puerta, "Ant colony optimization for learning Bayesian networks," *International Journal of Approximate Reasoning*, vol. 31, no. 3, pp.291–311, 2002.
[26] L. M. de Campos, J. A. Gamez, and J. M. Puerta, "Learning bayesian networks by ant colony optimisation: Searching in the space of orderings," *Mathware and Soft Computing*, vol. 9, no. 2–3, pp. 251–268, 2002.
[27] A. Shmygelska and H. H. Hoos, "An ant colony optimisation algorithm for the 2d and 3d hydrophobic polar protein folding problem," *BMC Bioinformatics*, vol. 6:30, 2005.
[28] O. Korb, T. Stützle, and T. E. Exner, "Application of ant colony optimization to structure-based drug design," in *Ant Colony Optimization and Swarm Intelligence, 5th International Workshop, ANTS 2006*, ser. LNCS, M. Dorigo *et al.*, Eds., vol. 4150. Springer Verlag, 2006, pp. 247–258.
[29] M. Dorigo, M. Birattari & T. Stützle, "Ant Colony Optimization – Artificial Ants as a Computacional Intelligence Technique", Techncal Report No. TR/IRIDIA/2006-023, September 2006.
[30] M. Mann, C. Smith, M. Rabbath, M. Edwards, S. Will & R. Backofen, "CPSP-web-tools: a server for 3D lattice protein studies", Bioinformatics Application Note, vol. 25 no. 5 2009, pages 676-677, doi: 10.1093/bioinformatics/btp034
[31] A. Shmygelska & H. H Hoos, "An ant colony optimization algorithm for the 2D and 3D hydrophobic polar protein folding problem", BMC Bioinformatics, 6:30, February 2005, doi: 10.1186/1471-2105-6-30
[32] Oakley MT, Barthel D, Bykov Y, Garibaldi JM, Burke EK, Krasnogor N & Hirst JD, "Search strategies in structural bioinformatics", Current Protein and Peptide Science, Vol.9, No.3, pp.260-274, 2008.
[33] http://folding.stanford.edu/Portuguese/Science, acedido em 3 de Novembro de 2015.
[34] http://biologia-geologia11a.blogspot.pt/2010/10/sintese-de-proteinas.html, acedido em 3 de Novembro de 2015.
[35] https://www.google.pt/search?q=transcri%C3%A7ao+proteinas&safe=off&biw=1745&bih=828&source=lnms&tbm=isch&sa=X&ved=0CAYQ_AUoAWoVChMI4cyYlYv0yAIVglYUCh1AMwrS#safe=off&tbm=isch&q=transcri%C3%A7ao+e+tradu%C3%A7%C3%A3o+proteinas&imgrc=V5pchLReF-eppM%3A, acedido em 3 de Novembro de 2015.
[36] http://zerpoii.opentronix.com/?p=953, acedido em 3 de Novembro de 2015.
[37] http://disciplinas.ist.utl.pt/qgeral/biomedica/proteinas-2.html, acedido em 3 de Novembro de 2015.
[38] P. F. N. Faísca, "O mistério da forma das proteínas", Gazeta da Física, vol 29, fasc. 3, pp.34-39, 2006.
[39] http://cftc.cii.fc.ul.pt/PRISMA/capitulos/capitulo4/modulo4/topico1.php, acedido em 3 de Novembro de 2015.
[40] http://repositorio.utfpr.edu.br/jspui/bitstream/1/112/1/CT_CPGEI_M_Scapin,%20Marcos%20Paulo_2005.pdf, acedido em 4, 18 de Novembro de 2015
[41] http://cftc.cii.fc.ul.pt/PRISMA/capitulos/capitulo4/modulo4/topico6.php, acedido em 3 de Novembro de 2015.
[42] http://www.fabriciobreve.com/material/cin/CIN-11-ACO.pdf, acedido em 16 de Novembro de 2015.
[43] http://www.ppgia.pucpr.br/~alekoe/IC/20042/private/monografias/Monografia-JulioZanoni.pdf, acedido em 16 de Novembro de 2015.
[44] http://www.ime.unicamp.br/~chico/mt852/formigas.pdf, acedido em 16 de Novembro de 2015.

[45] http://www.inf.ufpr.br/andre/Disciplinas/BSc/CI065/michel/Intro/intro.html, acedido em 16 de Novembro de 2015.
[46] Dorigo, M. & Stützle T., "Ant Colony Optimization", 2004, A Bradford Book, The MIT Press, Cambridge -Massachusetts, London – England.
[47] http://folding.stanford.edu/, acedido em 5 de Dezembro de 2015.
[48] http://www.ibm.com/developerworks/library/l-bluegene/, acedido em 5 de Dezembro de 2015.

Anexos

Anexo A – Proposta de Projecto

Anexo A

DEPARTAMENTO DE ENGENHARIA
INFORMÁTICA E DE SISTEMAS

INSTITUTO SUPERIOR DE
ENGENHARIA DE COIMBRA

PROPOSTA DE PROJECTO

MESTRADO em INFORMÁTICA E SISTEMAS
Ramo de Desenvolvimento de Software

Ano Lectivo de 2014/2015

TEMA

Previsão da Estrutura de Proteínas com Modelos HP

SUMÁRIO

A previsão da estrutura tridimensional das proteínas, tomando como ponto de partida a sua sequência de aminoácidos, é um dos problemas mais importantes da área da bioinformática. Os modelos HP simplificam algumas das restrições associadas ao processo de *folding* de uma proteína e viabilizam a obtenção de soluções aproximadas num curto espaço de tempo.

1. ÂMBITO

A construção de modelos HP é feita através da utilização de métodos de optimização estocásticos. Estes métodos procuram, de forma iterativa, os modelos que melhor se adequam a uma situação concreta. A construção é efectuada através da colocação de aminoácidos numa grelha discreta com 2 ou 3 dimensões. O trabalho a desenvolver neste projecto consiste no desenvolvimento e aplicação de algoritmos de optimização a problemas de previsão com modelos HP e na construção de uma plataforma Web que permita disponibilizar à comunidade as ferramentas desenvolvidas.

2. OBJECTIVOS

O objectivo principal deste projecto é desenvolver e testar algoritmos de optimização baseados em colónias de formigas para resolver problemas de construção de modelos HP. Será dedicada especial atenção à fase de construção de soluções e ao modo como os métodos de optimização se comportam em espaços com diferentes topologias. A avaliação do desempenho dos algoritmos será feita com recurso a instâncias HP disponíveis em *benchmarks* habitualmente utilizados pela comunidade científica.

Um objectivo complementar do projecto é a disponibilização de uma aplicação Web que disponibilize à comunidade as ferramentas desenvolvidas.

3. PROGRAMA DE TRABALHOS

1. Revisão bibliográfica do problema da previsão da estrutura de proteínas, dos modelos HP e dos métodos de optimização baseados em colónias de formigas
2. Concepção e desenvolvimento dos algoritmos de optimização
3. Testes e análise de resultados
4. Concepção e desenvolvimento da aplicação Web
5. Validação da aplicação Web
6. Escrita da documentação final

4. CALENDARIZAÇÃO DAS TAREFAS

As Tarefas acima descritas serão executadas de acordo com a seguinte calendarização:

	Meses									
Tarefas		N	N+1	N+2	N+3	N+4	N+5	N+6	N+7	N+8
T1										
T2										
T3										
T4										
T5										
T6										
Metas	INI									

5. RESULTADOS

Os resultados do estágio serão consubstanciados num conjunto de documentos a elaborar pelo estagiário de acordo com o seguinte plano:

- Relatórios/Sumários das reuniões periódicas
- Dissertação de Mestrado

6. LOCAL DE TRABALHO

DEIS-ISEC

7. METODOLOGIA

8. ORIENTAÇÃO

Francisco Pereira, Professor Adjunto
xico@isec.pt

9. CARACTERIZAÇÃO

- Data de início: Outubro 2014
- Data de fim: Julho 2015