UNIVERSIDADE TÉCNICA DE LISBOA

INSTITUTO SUPERIOR TÉCNICO

Sistema de gestão da informação dos mecanismos de regulação genómica do organismo *Saccharomyces cerevisiae*

Pedro Tiago Gonçalves Monteiro

(Licenciado)

Dissertação para Obtenção do Grau de Mestre em
Engenharia Informática e de Computadores

Orientador Científico:	Doutor Arlindo Manuel Limede de Oliveira
Co-Orientadora Científica:	Doutora Ana Teresa Correia de Freitas
Presidente do Júri:	Doutora Isabel Maria de Sá-Correia Leite de Almeida
Vogais:	Doutor Mário Jorge Costa Gaspar da Silva
	Doutor Arlindo Manuel Limede de Oliveira
	Doutora Ana Teresa Correia de Freitas

Lisboa, 20 de Março de 2005

UNIVERSIDADE TÉCNICA DE LISBOA

INSTITUTO SUPERIOR TÉCNICO

Sistema de gestão da informação dos mecanismos de regulação genómica do organismo *Saccharomyces cerevisiae*

Pedro Tiago Gonçalves Monteiro

(Licenciado)

Dissertação para Obtenção do Grau de Mestre em
Engenharia Informática e de Computadores

Orientador Científico:	Doutor Arlindo Manuel Limede de Oliveira
Co-Orientadora Científica:	Doutora Ana Teresa Correia de Freitas
Presidente do Júri:	Doutora Isabel Maria de Sá-Correia Leite de Almeida
Vogais:	Doutor Mário Jorge Costa Gaspar da Silva
	Doutor Arlindo Manuel Limede de Oliveira
	Doutora Ana Teresa Correia de Freitas

Lisboa, 20 de Março de 2005

O trabalho subjacente à presente dissertação foi realizado sob a orientação do

Professor Arlindo Manuel Limede de Oliveira

Professor Associado (c/ Agregação) do Departamento de Engenharia Informática e de Computadores do Instituto Superior Técnico, da Universidade Técnica de Lisboa

e sob co-orientação da

Professora Ana Teresa Correia de Freitas

Professora Auxiliar do Departamento de Engenharia Electrotécnica e de Computadores do Instituto Superior Técnico, da Universidade Técnica de Lisboa

Resumo

Após a sequenciação dos genomas de diversos organismos, passou-se à fase de anotação dos genes, tendo muita desta informação ficado disponível para ser processada e transformada em conhecimento.

Neste contexto, é especialmente importante o estudo dos mecanismos de interacções entre genes, ou seja, o estudo das redes de regulação genética.

Este trabalho tem como objectivo desenvolver uma plataforma que suporte o estudo destas redes, esperando-se que venha a constituir um repositório integrado de toda a informação relevante para os fenómenos de regulação genómica no organismo *Saccharomyces cerevisiae*. A plataforma inclui uma base de dados de suporte à informação já adquirida, interfaces para utilizadores finais e para administradores e curadores da informação. Inclui também conectores para implementações de algoritmos de análise e suporte à descoberta de conhecimento.

Keywords: Bases de dados, *Saccharomyces cerevisiae*, redes de regulação, genes, factores de transcrição, *consensus*

Abstract

After the sequencing of the genome of a number of organisms, the main challenge resides in the interpretation of the large amounts of data generated.

In this context, is it specially relevant the study of the mechanisms of interaction between genes, that is, the study of gene regulatory mechanisms.

This work consists in the development of a platform that supports the study of these networks, and is expected to evolve into an integrated repository of all relevant information for the phenomena of genetic regulation in the organism *Saccharomyces cerevisiae*. The platform includes a database for storing already known information, interfaces for final users and for administrators and curators. It also includes connectors with implementations of algorithms that will be used to support the knowledge discovery process.

Keywords: Databases, *Saccharomyces cerevisiae*, regulatory networks, genes, transcription factors, *consensus*

Agradecimentos

Quero agradecer aos meus orientadores, Prof. Arlindo Oliveira e Prof. Ana Teresa Freitas pela forma como orientaram este trabalho, e pelo trabalho de revisão desta tese.

Gostaria também de agradecer a todas as pessoas do grupo de Ciências Biológicas do Departamento de Química do IST, pela cooperação no desenvolvimento deste sistema. Em especial, à Prof. Isabel Sá-Correia pelo seu sentido crítico, e ao Miguel Teixeira pelo acompanhamento constante.

Gostaria também de agradecer aos meus colegas do grupo ALGOS, em especial à Ana, Óscar, Orlando e Miguel pelas sugestões e críticas construtivas, e ao Nuno pelo trabalho de revisão desta tese.

Por último, gostaria de agradecer à minha família, pelo apoio e suporte demonstrados ao longo de todos estes anos.

Glossário

ADN - Molécula de Ácido desoxirribonucleico composta por duas cadeias de nucleótidos formando uma dupla hélice. O ADN transporta a informação genética necessária para a organização e funcionamento de uma célula.

Anotação de genes - A informação relativa às entidades envolvidas no estudo de genes, em que condições foi obtida e por quem, guardada num repositório público para consulta posterior.

Anti-codão - Sequência de três nucleótidos pertencente ao tARN que é complementar a um codão no mARN.

ARN - Molécula de Ácido ribonucleico. Esta molécula é utilizada durante a síntese de proteínas.

ARN mensageiro (mARN) - Molécula de ARN que tem como objectivo transportar a informação contida no ADN até ao processo de tradução. Esta molécula é transcrita a partir da molécula de ADN, sofre um processo de maturação onde são retirados todos os nucleótidos não codificantes, e é traduzida em aminoácidos no ribossoma.

ARN polimerase - Enzima responsável pela síntese de uma nova molécula de ARN.

ARN ribossómico (rARN) - Constituinte dos ribossomas.

ARN transferência (tARN) - Molécula de ARN que transporta os aminoácidos para o ribossoma para a síntese de um polipéptido. Durante a tradução quando o anti-codão emparelhar com um codão pertencente ao mARN, o aminoácido transportado é inserido na cadeia polipeptidica.

Codão - Sequência específica de três nucleótidos no mARN. Durante o processo de tradução têm uma correspondência para um dos vinte aminoácidos ou para o sinal de terminação do processo de tradução.

Eucariota - Organismo constituído por uma ou mais células. As células deste organismo contêm ainda uma membrana no seu interior formando um núcleo.

Nucleótido - Sub-unidade constituinte do ADN e ARN. Cada nucleótido é constituído por uma base azotada (Adenina, Timina, Citosina e Guanina no ADN; Adenina, Uracilo, Citosina e Guanina no ARN), uma molécula de açúcar e um grupo fosfato.

Open Reading Frame (ORF[1]) - Sequência de nucleótidos flanqueada por um codão de iniciação e um codão de finalização, dentro de uma determinada janela de leitura. Nem todas as ORF contêm sequências codificantes. O facto de aparecer um codão de iniciação ou finalização pode ser fruto do mero acaso ao longo do genoma.

Retro-transposão - Transposão que foi criado a partir de transcrição inversa.

Ribossoma - Estrutura da célula onde o mARN é traduzido durante a síntese de proteínas. Esta estrutura divide-se em duas subunidades: uma grande (50S) e uma menor (30S).

Tradução - Processo do Dogma Central da Biologia, em que uma cadeia de mARN é traduzida num polipéptido. Corresponde ao último processo da síntese de uma proteína.

Transcrição - Processo do Dogma Central da Biologia, em que uma cadeia de ADN é transcrita numa cadeia de mARN.

Transposão - Sequência de ADN que é flanqueado por sequências repetidas, com capacidade de se mover ao longo da sequência de ADN.

***Splicing* Alternativo** - Etapa durante a maturação do mARN em que os intrões são removidos e é possível um rearranjo ou uma selecção dos exões existentes nesse gene. Esta etapa permite a um gene codificar mais do que uma única proteína.

[1]Em Português, Grelha de leitura aberta

Conteúdo

Lista de Figuras

Lista de código SQL

Capítulo 1

Introdução

1.1 Motivação

Até à década de 90, o principal objectivo dos projectos na área da genómica consistia na sequenciação de genomas dos mais diversos organismos. Todos os dias, sequências com milhões de bases de ácido desoxirribonucleico (ADN) foram, e continuam a ser, armazenadas em grandes bases de dados. Apesar da sequenciação de genomas continuar a produzir um enorme volume de informação, a comunidade científica passou a dar uma maior ênfase à transformação desses dados em conhecimento. A necessidade de software cada vez mais sofisticado, bem como de novos algoritmos levou ao aparecimento de uma nova área de interligação entre a biologia molecular e as ciências de computação, a Bioinformática. Este termo foi criado em meados dos anos 80 e originalmente, referia-se à manipulação e análise de sequências recorrendo ao uso de computadores. Actualmente, refere-se à aplicação das ciências da computação na aquisição, manipulação e análise de todo o tipo de informação biológica.

A expressão de um gene específico está dependente da presença de determinadas proteínas na célula, os factores de transcrição. Por sua vez, estas proteínas são o resultado da expressão de outros genes. As relações gene $\rightarrow$ proteína $\rightarrow$ gene são normalmente designadas de redes de regulação genética.

O conhecimento destas redes de regulação é de extrema importância na investigação em Biologia Molecular e Medicina, tendo aplicações tão variadas como a análise do ciclo de vida das células, o estudo de doenças hereditárias, a evolução do cancro ou o desenvolvimento de terapêuticas.

O sistema desenvolvido nesta tese consiste numa base de dados de factores de transcrição. Este sistema permite identificar mecanismos de regulação e visualizar potenciais factores de transcrição, entre outras funcionalidades. Este sistema está a ser desenvolvido em estreita colaboração com o grupo de Ciências Biológicas do IST, e concentra muita da informação referente ao organismo *Saccharomyces cerevisiae*, que é uma levedura. Este organismo é o eucariota mais simples, sendo amplamente estudado pela comunidade científica.

1.2 Contribuições e Objectivos desta Tese

O principal objectivo desta tese é o desenvolvimento de uma base de dados que relacione a informação genómica associada à regulação de genes, por forma a permitir a análise da influência, directa ou indirecta, de um determinado gene, na regulação de outros genes. De uma forma mais lata, pretendemos criar uma ferramenta que auxilie na identificação das redes de regulação genéticas do organismo *Saccharomyces cerevisiae*.

Existem, actualmente, várias bases de dados que providenciam parcialmente esta informação. No entanto, parte destas bases de dados não estão actualizadas, porque deixaram de ser mantidas, como é o caso da *SCPD* [1]. Outras, quando começaram a ter um interesse público significativo tornaram-se fechadas, como é o caso da *YPD* [2] que se tornou propriedade da *Incyte Corporation*, que apresenta actualmente restrições de acesso. Para o organismo *Saccharomyces cerevisiae* existe a base de dados oficial da comunidade que o estuda, a *Saccharomyces Genome Database (SGD)* [3]. Esta disponibiliza praticamente toda a informação obtida pelos grupos de investigação, mas não apresenta um cruzamento de informação adequado para o problema da identificação ou inferência de redes de regulação genéticas.

Pretende-se também, com este trabalho, a integração de muitos dos dados existentes noutras bases de dados. Assim, o sistema desenvolvido tem os seguintes objectivos:

- permitir o acesso generalizado de todo o público, independentemente do seu carácter público ou privado e qualquer que seja o seu interesse;
- disponibilizar informação relativa aos promotores de genes, tal como a *SCPD* [1];
- disponibilizar informação relativa a factores de transcrição, as suas zonas de *consensus* e a lista de genes regulados, tal como a *TRANSFAC* [4] e a *SCPD* [1];

- permitir a visualização dos genes potencialmente regulados por um determinado factor de transcrição;
- integrar as hierarquias de terminologias do *Gene Ontology Consortium [5]*, permitindo assim dar um contexto semântico às pesquisas efectuadas.

1.3 Organização do documento

Este documento está estruturado da seguinte forma:

No glossário são apresentados alguns dos termos referentes à Biologia Molecular.

No capítulo 1 é efectuada uma introdução do sistema desenvolvido e apresentadas as contribuições que o mesmo oferece.

No capítulo 2 são introduzidos alguns conceitos básicos de Biologia Molecular essenciais à compreensão dos processos relativos ao Dogma Central da Biologia, ou seja, os processos necessários à expressão dos genes. É também feita a ponte entre estes processos biológicos e a necessidade da informática na análise e compreensão dos mesmos.

No capítulo 3 é descrito o processo de identificação e definição dos conceitos biológicos de forma a serem introduzidos na base de dados. São também descritas as três ontologias desenvolvidas pelo *Gene Ontology Consortium* e a forma como estas foram utilizadas no sistema desenvolvido.

No capítulo 4 é descrita a arquitectura do sistema, dando particular importância aos métodos utilizados na extracção, tratamento e carregamento da informação para a base de dados.

No capítulo 5 são descritas as funcionalidades implementadas.

No capítulo 6 é efectuada a avaliação do sistema, demonstrando a importância do desenvolvimento de um sistema desta natureza.

No capítulo 7 são apresentadas as conclusões e perspectivado o trabalho futuro.

Capítulo 2

Conceitos de Biologia Molecular

A Biologia Molecular dedica-se ao estudo das interacções entre os vários sistemas da célula, incluindo as relações entre o ácido desoxiribonucleico (ADN), ácido ribonucleico (ARN) e síntese de proteínas, tendo como um dos objectivos principais explicar como estas interacções são reguladas.

Neste capítulo são apresentados alguns dos conceitos básicos da área da Biologia Molecular necessários à compreensão do processo de transcrição de genes, descrito na secção 2.5. Pretende-se com a apresentação destes conceitos facilitar a compreensão da importância do trabalho desenvolvido para a investigação da área da Biologia Molecular.

2.1 Molécula de ADN

O ADN é a molécula base do material genético encontrado em todas as células, contendo a informação necessária para controlar os mecanismos celulares. É através do ADN que a informação genética é passada de geração em geração. A molécula de ADN é composta por duas cadeias de nucleótidos unidas em dupla hélice (Figura 2.1). Um nucleótido é uma molécula composta por uma pentose, um grupo fosfato e uma base azotada. No ADN, podem ser encontrados quatro tipos de nucleótidos, diferindo a sua composição apenas na base azotada.

A cada tipo de nucleótido está associada uma letra que é a abreviatura da sua base azotada: Adenina (A), Guanina (G), Citosina (C) e Timina (T). Estas bases azotadas encontram-se no interior da dupla hélice, sendo responsáveis pelo emparelhamento das duas cadeias do ADN. O emparelhamento das bases das duas cadeias de ADN é muito específico: a Adenina

emparelha com a Timina e a Guanina emparelha com a Citosina. A descoberta da estrutura tri-dimensional do ADN deve-se a J. Watson e F. Crick [6] em 1953.

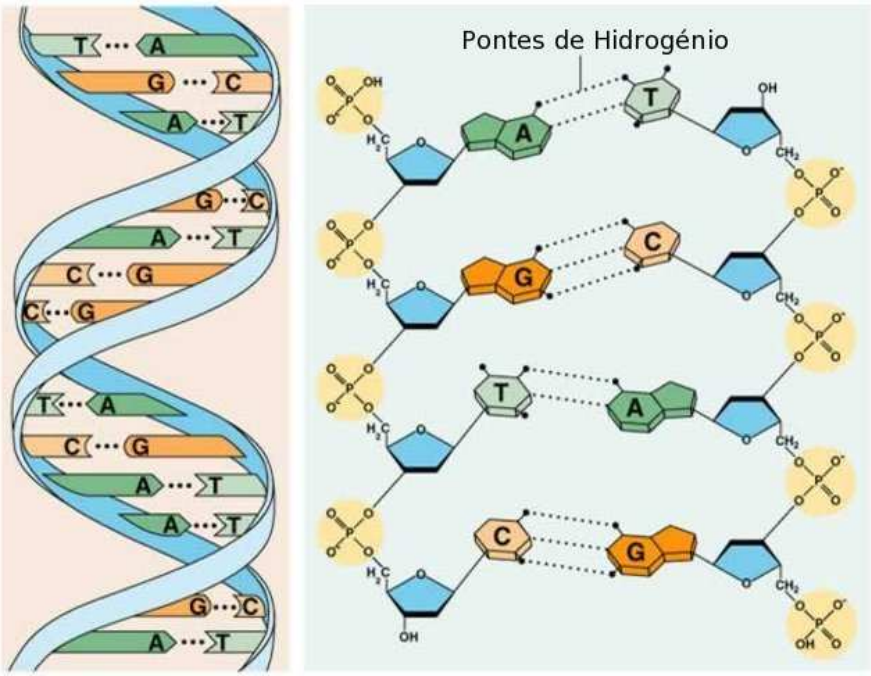


Figura 2.1: Estrutura de dupla hélice do ADN.

A estrutura em dupla hélice, em conjunto com a limitação de emparelhamento das suas bases azotadas, impõe que a ordem da sequência de bases de uma cadeia defina a ordem da outra cadeia. Diz-se, por isso, que as cadeias são complementares.

2.2 Molécula de ARN

O Ácido Ribonucleico (ARN) é um ácido nucleico semelhante ao ADN. No entanto, existem três diferenças principais entre estes dois ácidos.

A primeira prende-se com o facto de os nucleótidos da molécula de ARN conterem o açúcar ribose ao contrário da desoxirribose (esta é a origem da diferença no nome das moléculas). A segunda diferença tem a ver com o facto de a molécula de ARN não conter a base azotada Timina (T). Em substituição desta base azotada, o ARN contém outra base azotada com o nome de Uracilo (U). A terceira diferença tem a ver com o facto de a estrutura da molécula de ARN não ser em dupla hélice. A molécula de ARN é constituída por uma cadeia simples

de nucleótidos.

2.2.1 ARN Mensageiro

O ARN Mensageiro (mARN) contém um cópia da informação genética contida no ADN, tendo como principal diferença a substituição da base azotada Timina pelo Uracilo. O mARN é sintetizado a partir do ADN aquando do processo de transcrição (ver secção 2.4.1). Nos organismos eucariotas, este processo ocorre dentro do núcleo da célula. O mARN resultante vai transportar a informação genética do núcleo para o citoplasma, onde irá ocorrer outro processo celular, a tradução (ver secção 2.4.3), que culmina com a síntese de um polipéptido, uma proteína. No caso dos organismos procariotas, não existe um núcleo individualizado e a síntese do mARN é imediatamente seguida pela síntese proteíca, ocorrendo quase em simultâneo.

2.2.2 ARN Transferência

A molécula do ARN Transferência (tARN), como se pode ver na Figura 2.2, é uma sequência de nucleótidos que estabelecem ligações entre si, formando uma estrutura cuja função é transporte dos aminoácidos correspondentes aos codões (ver glossário) lidos do mARN.

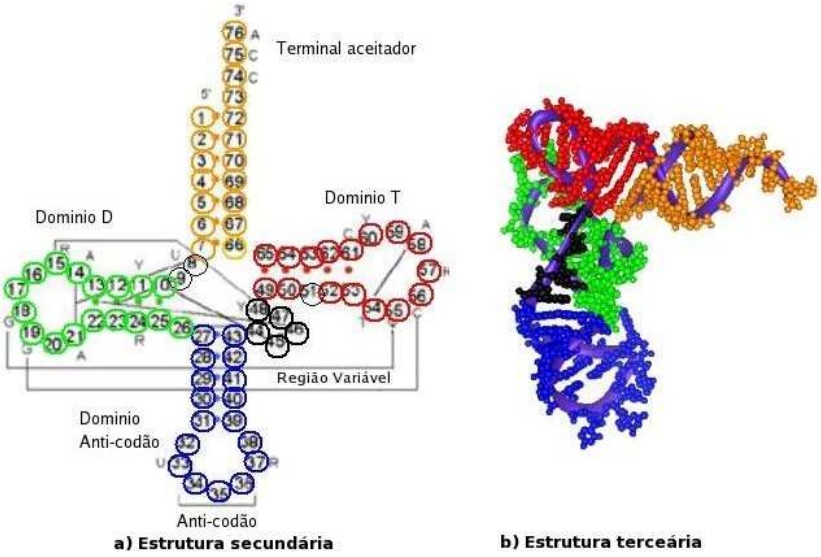


Figura 2.2: ARN Transferência.

A sequência de três nucleótidos do tARN que emparelha com a sequência de três nucleótidos do mARN tem o nome de anti-codão (ver glossário).

Existe um tARN específico para o transporte de cada aminoácido. No entanto, o local de ligação entre qualquer tARN e o aminoácido correspondente é composto pela mesma sequência de três nucleótidos, AAC. A proteína sintetizada é composta pelo conjunto de aminoácidos transportados pelos tARN até ao ribossoma.

2.2.3 ARN Ribossómico

Outra forma de ARN é o ARN Ribossómico (rARN). Este ARN é o maior constituinte dos ribossomas (ver glossário). Os ribossomas são organitos celulares onde ocorre a síntese proteíca, ou seja, onde o mARN é lido e traduzido para dar origem a uma proteína, como ilustrado na Figura 2.3. São constituídos por duas sub-unidades de diferentes tamanhos. A sub-unidade maior contém o local de ligação do tARN carregado com o aminoácido respectivo. A menor contém o local de ligação para o mARN. Assim, a sub-unidade mais pequena do ribossoma fixa o mARN, enquanto a sub-unidade maior faz o emparelhamento do codão com o anti-codão do tARN. Para cada aminoácido que vai sendo transportado pelo tARN, o ribossoma promove a ligação de ligações peptídicas para ligar esse aminoácido aos anteriores.

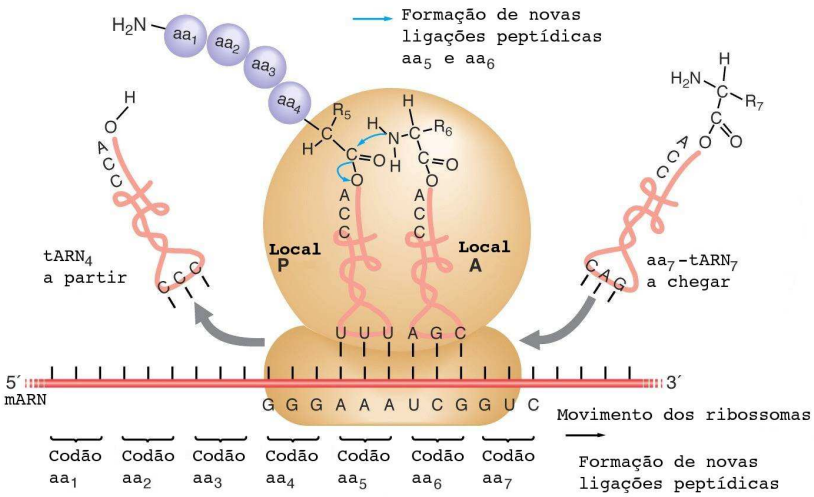


Figura 2.3: Tradução do mARN - Ribossoma a efectuar a síntese proteíca.

2.3 Estrutura dos Genes (em eucariotas)

O código genético de uma determinada sequência de ADN é definido pela sequência de bases na cadeia de nucleótidos. A ordem pela qual as quatro bases aparecem ao longo de cadeia de ADN é, portanto, crítica para a célula, correspondendo às instruções do programa genético dos organismos. A cadeia de ADN é formada por zonas codificantes, os genes, e zonas não codificantes, zonas intergénicas, como apresentado na Figura 2.4.

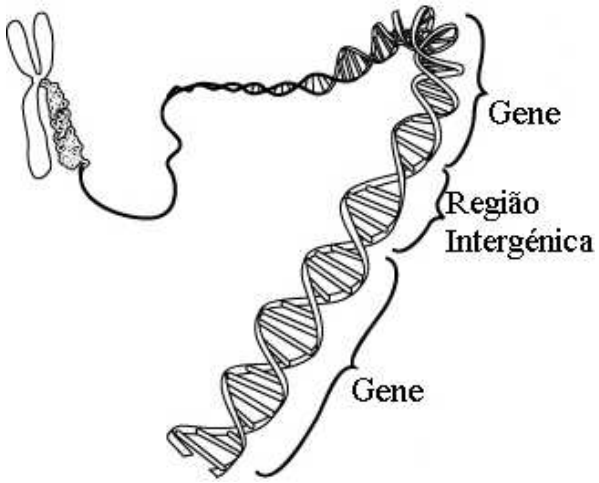


Figura 2.4: Estrutura dos genes.

A cada gene está associada uma região promotora situada geralmente a montante deste, servindo de ligação às proteínas que iniciam o processo de transcrição, como apresentado na Figura 2.5. A seguir à região promotora o gene é composto por uma sequência de início da transcrição e por uma região que, apesar de transcrita, não será traduzida, denominada de UTR[1]. A seguir a esta região existe o sinal de início de tradução, sinal que dá início à síntese de proteínas (ver secção 2.4.3).

Internamente os genes são constituídos por sequências codificantes, os exões, intercaladas por sequências não codificantes, os intrões (Figura 2.5). Na região a jusante, temos o sinal de terminação da tradução, e o sinal de terminação de transcrição, separados por uma região não traduzida.

Os genes contêm as instruções para criar as milhares de proteínas encontradas numa célula. O conjunto de genes que constituem a informação genómica de um organismo tem o nome de

[1]Do inglês, *Untranslated Region.*

genoma.

Sendo os genes os portadores da informação genética essencial para a criação das proteínas, convém perceber qual é a relação entre a linguagem do ADN, os nucleótidos, e a das proteínas, os aminoácidos.

2.4 Expressão dos Genes

As proteínas são constituídas por aminoácidos que, tal como os nucleótidos na sequência de ADN, estão ordenados numa sequência linear. Conhecem-se vinte aminoácidos comuns a todos os organismos. A ordenação dos aminoácidos numa molécula proteíca confere-lhe características e funções biológicas específicas. A alteração de um aminoácido numa sequência pode conduzir a uma modificação na estrutura e função biológica dessa molécula.

A informação para a ordenação dos aminoácidos está contida no ADN sob a forma de um código que reside na sequência das bases azotadas da molécula. O processo biológico para a síntese de uma proteína resume-se, basicamente, à conversão da informação contida numa sequência de nucleótidos de ADN para a sequência de aminoácidos da proteína.

O código genético corresponde ao dicionário que a célula utiliza para traduzir a linguagem do ADN em linguagem proteíca. Cada três nucleótidos constituiem uma palavra, codão, que determina um aminoácido. Embora vários codões codifiquem o mesmo aminoácido, o mesmo codão nunca codifica aminoácidos diferentes. O codão ATG tem uma dupla função: codifica o aminoácido meteonina e é um codão de iniciação[2] da síntese proteíca. Os codões TAA, TAG e TGA não designam aminoácidos e representam sinais de fim de síntese, chamando-se codões de finalização[3].

No processo biológico em que a partir de uma sequência de ADN é gerada uma proteína (Figura 2.5), existe uma molécula, o Ácido Ribonucleico (ARN), que actua como intermediária. Este fluxo de informação genética, designado de Dogma Central da Biologia Molecular [7], envolve quatro etapas: a transcrição, a maturação, a tradução e a replicação.

[2]Em Inglês, *start codon.*
[3]Em Inglês, *stop codon.*

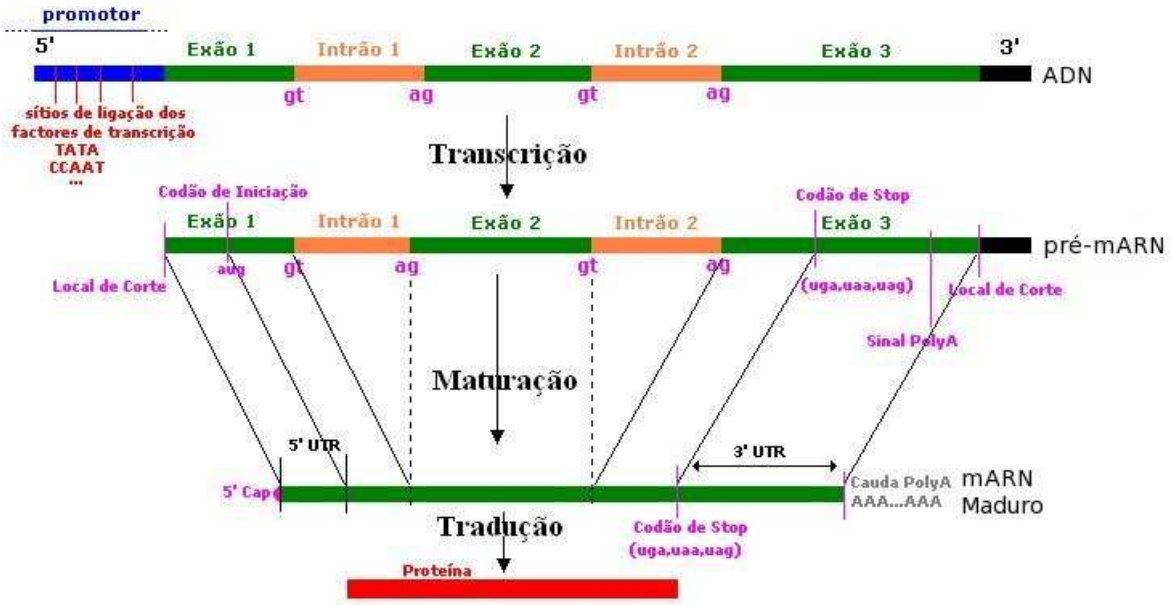


Figura 2.5: Dogma Central da Biologia Molecular - do ADN às Proteínas.

2.4.1 Transcrição

O primeiro passo, a transcrição, tem como objectivo a síntese de ARN. Neste passo é feita uma cópia complementar da cadeia de ADN, sendo a base azotada Timina (T) substituída por uma outra base, o Uracilo (U) que, como já foi anteriormente referido, apenas existe na molécula de ARN.

A transcrição de um segmento de ADN forma um ARN mensageiro preliminar, o pre-mARN. Esta transcrição inicia-se sempre na extremidade a montante do gene, designada de terminal 5' da cadeia de sentido directo[4]. A ARN Polimerase II sintetiza ARN no sentido 5' → 3', que é igual à cadeia de ADN no sentido directo, servindo-se da cadeia de sentido inverso[5] como molde, terminando na extremidade a jusante do gene, designada de terminal 3'. Devido ao facto do ADN ser constituído por duas cadeias complementares, o terminal 5' de uma cadeia corresponde ao terminal 3' da outra.

A transcrição inicia-se com a ligação da ARN Polimerase II à região promotora do gene. Para além da ARN Polimerase II, diversas proteínas ligam-se num complexo para dar início à transcrição. As proteínas deste complexo denominam-se factores de transcrição. Alguns

[4]Em Inglês, *forward strand.*
[5]Em Inglês, *reverse strand.*

factores de transcrição ligam-se a outros factores de transcrição formando um complexo proteico. No entanto, alguns destes factores de transcrição reconhecem sequências específicas na região promotora do gene denominadas de zonas de *consensus*.

Depois do complexo proteico dar início à transcrição, o ARN vai sendo sintetizado até ser encontrado o sinal de fim de transcrição, sinal este que faz com que o complexo se dissocie e se liberte do ADN terminando a transcrição.

2.4.2 Maturação

Após a transcrição, o pre-mARN é sujeito a um processo de maturação onde são retirados os intrões, havendo posteriormente a união dos exões. Esta remoção é denominada de *splicing*.

Ainda nesta fase, ambas as extremidades da molécula de mARN sofrem algumas alterações. No terminal 5' é adicionado um terminal CAP[6], ou seja, é adicionada uma base guanina com um grupo metil, e no terminal 3' é adicionada uma cauda *poly-A* composta por várias bases de adenina. Esta cauda *poly-A* está relacionada com o controlo do tempo de vida do mARN no citoplasma.

Estas transformações conduzem à formação de um ARN mensageiro maduro, o mARN. Esta molécula é então transportada do núcleo para o citoplasma levando a informação para a síntese de uma proteína. A informação necessária está codificada nos codões da sequência de mARN.

2.4.3 Tradução

A terceira etapa do Dogma Central da Biologia consiste na tradução dos codões da sequência de mARN em aminoácidos de acordo com o código genético. A tradução desencadeia-se a partir da extremidade 5' da cadeia de mARN e começa no codão de iniciação, ATG. Os codões vão sendo sucessivamente traduzidos em aminoácidos e a síntese termina quando se chega a um dos codões de finalização.

Visto que cada codão é composto por três nucleótidos e existem quatro nucleótidos, ficamos assim com $4^3 = 64$ possíveis combinações de três nucleótidos. No entanto, existem apenas 20 aminoácidos, o que significa que existem aminoácidos que são codificados por mais do que

[6]Em Inglês, *capping*. É este terminal que indica o início de tradução do mARN numa proteína

um codão. Devido a este facto diz-se que o código genético é degenerado. Na Figura 2.6, podemos ver a correspondência entre os codões e os aminoácidos.

Primeira Posição	Segunda Posição: U				Segunda Posição: C				Segunda Posição: A				Segunda Posição: G				Terceira Posição
	Código	Aminoácido	Código	Aminoácido	Código	Aminoácido	Código	Aminoácido	Código	Aminoácido	Código	Aminoácido					
U	UUU	phe	UCU	ser	UAU	tyr	UGU	cys	U								
	UUC		UCC		UAC		UGC		C								
	UUA	leu	UCA		UAA	STOP	UGA	STOP	A								
	UUG		UCG		UAG	STOP	UGG	trp	G								
C	CUU	leu	CCU	pro	CAU	his	CGU	arg	U								
	CUC		CCC		CAC		CGC		C								
	CUA		CCA		CAA	gin	CGA		A								
	CUG		CCG		CAG		CGG		G								
A	AUU	ile	ACU	thr	AAU	asn	AGU	ser	U								
	AUC		ACC		AAC		AGC		C								
	AUA		ACA		AAA	lys	AGA	arg	A								
	AUG	met	ACG		AAG		AGG		G								
G	GUU	val	GCU	ala	GAU	asp	GGU	gly	U								
	GUC		GCC		GAC		GGC		C								
	GUA		GCA		GAA	glu	GGA		A								
	GUG		GCG		GAG		GGG		G								

Figura 2.6: Correspondência entre os codões e os aminoácidos.

Este código degenerado serve como um mecanismo de minimização da propagação de erros do ADN até às proteínas. Assim, se a sequência de ADN sofrer uma mutação num determinado nucleótido, o mARN transcripto propagará essa mutação para fora do núcleo até aos ribossomas, onde a proteína será sintetizada. No entanto, aquando da tradução do aminoácido o erro poderá ser evitado pelo facto de o codão poder ser traduzido no mesmo aminoácido. Pode ser observado que os codões que sintetizam o mesmo aminoácido, normalmente diferem apenas no terceiro nucleótido.

Após a etapa da tradução obtem-se como produto final as proteínas, que são as unidades funcionais da célula. Sem proteínas nenhum processo celular poderia ocorrer.

2.4.4 Replicação

Esta quarta etapa não ocorre durante o processo de síntese de proteínas. No entanto, é aqui apresentado por fazer parte do Dogma Central da Biologia. Este passo ocorre aquando da divisão das células, com o objectivo de passar a informação genética de uma célula mãe para uma célula filha, ou seja, copiar o ADN existente na célula mãe para a célula filha.

Nesta etapa, complexos proteicos ligam-se a cada uma das cadeias de ADN separando-as, enquanto outros efectuam a cópia do ADN. Assim, cada nova cadeia sintetizada é uma

cópia complementar da cadeira original, originando duas cadeias duplas, conforme ilustrado na Figura 2.7.

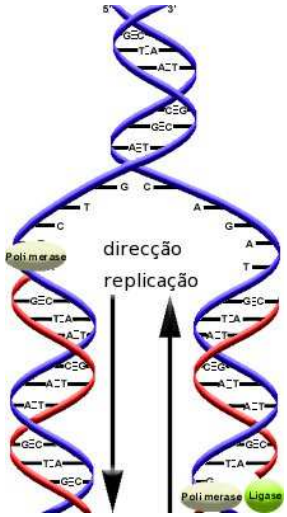


Figura 2.7: Replicação das duas cadeias complementares de ADN.

2.5 Regulação de genes

Utilizando os conceitos da Biologia Molecular descritos nas secções anteriores, serão introduzidos nesta secção os conceitos de regulação de genes, auto-regulação e redes de regulação.

As proteínas vão constituir os mais variados tipos de unidades funcionais da célula, actuando na degradação de nutrientes na célula, no transporte de entidades celulares ou até mesmo no suporte à síntese de novas proteínas. No caso de servirem de suporte à síntese de novas proteínas, uma das suas actividades específicas é regular (activando ou inibindo) o processo de transcrição. Estas proteínas, denominadas de factores de transcrição, contêm um domínio que reconhece uma sequência específica de nucleótidos no promotor do gene alvo, chamada zona de *consensus*.

É normal encontrar factores de transcrição que têm associados mais de uma zona de *consensus*, ou seja, que reconhecem mais do que uma sequência de nucleótidos na região promotora de um gene. O facto de reconhecerem mais do que uma única sequência de nucleótidos pode eventualmente reflectir um aumento do número de genes alvo da sua regulação.

Apesar de existirem vários mecanismos de *regulação* da expressão de um gene, um dos

mais importantes ocorre ao nível da transcrição. Durante a etapa da transcrição os factores de transcrição ligam-se à região promotora do gene, activando ou reprimindo a transcrição desse gene.

Existem por vezes factores de transcrição que regulam os próprios genes que lhe deram origem. Isto significa que um gene é transcrito e traduzido numa determinada proteína, e que esta proteína tem como função ligar-se à região promotora de genes, incluindo a do próprio gene que lhe deu origem. A este processo dá-se o nome de *auto-regulação*.

Existem ainda os conceitos de relação de activação e relação de repressão. Estes conceitos resultam do facto da regulação de genes poder ocorrer de duas formas distintas. Um factor de transcrição pode ter uma *relação de activação* com um determinado gene, o que significa que a ligação desse factor de transcrição ao promotor do gene vai influenciar positivamente a expressão desse gene. Por outro lado, um outro factor de transcrição pode ter uma *relação de repressão* com esse gene o que significa que a ligação desse factor de transcrição ao promotor do gene vai influenciar negativamente a sua expressão.

Existem factores de transcrição que podem ter ambas as funções, o que significa que, num determinado processo celular podem activar um conjunto de genes e noutro processo podem reprimir outros genes.

Ao conjunto das relações, de repressão e de activação, dá-se o nome de *redes de regulação*. Fazendo a ponte para a área da informática, mais precisamente para a área de algoritmos, estas redes podem ser modeladas utilizando uma estrutura de dados designada de *grafo*. O grafo vai representar a rede de regulação, indicando que a ligação de um factor de transcrição no promotor de um determinado gene vai influenciar a expressão deste.

Como podemos ver na Figura 2.8, podem existir no grafo nós ligados a si próprios (genes auto-regulados), nós ligados a vários outros (genes que codificam factores de transcrição que regulam múltiplos genes), e ainda múltiplos nós ligados ao mesmo nó (vários genes que codificam factores de transcrição que regulam um mesmo gene). O conhecimento destas redes tem várias aplicações. A organização dos milhares de genes de um organismo numa hierarquia, permite-nos identificar que repercussões tem a inactivação ou sobre-expressão de um determinado gene nos processos em que este está envolvido.

Por exemplo, a identificação das várias interacções entre os genes no estudo de determinada doença, permite, através do conhecimento da função celular dos produtos dos genes,

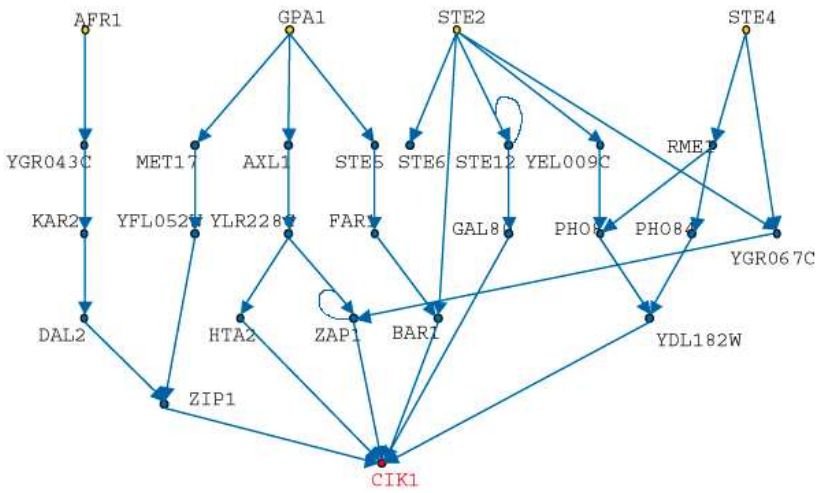


Figura 2.8: Exemplo de uma rede de regulação de genes.

prever as consequências dessa doença. O conhecimento de quais os genes essenciais para o desenvolvimento de uma determinada doença e de quais as consequências da sua inactivação, é de extrema importância.

2.5.1 Regulações documentadas vs. potenciais

Actualmente, as regulações de genes são anotadas através de experiências laboratoriais em que os biólogos verificam a influência de um ou mais factores de transcrição na expressão de um ou mais genes. Posteriormente, quando se tem conhecimento que um determinado factor de transcrição se liga ao promotor de determinado gene, tenta-se determinar a zona de *consensus*, ou seja, a sequência de nucleótidos na região promotora do gene a que o factor de transcrição se liga. Depois de conhecida esta sequência, esta é inserida numa base de dados, bem como a informação relativa ao factor de transcrição correspondente. Esta regulação é a partir deste momento uma regulação documentada.

A descoberta de novas regulações de genes é um desafio apresentado não só aos biólogos, mas também aos informáticos. Devido à grande quantidade de genes e factores de transcrição existentes, seria muito difícil para os biólogos conseguirem laboratorialmente verificar todas as combinações possíveis entre factores de transcrição e genes. Por outro lado, a computação pode, através de emparelhamento de cadeias de caracteres, verificar todas as possíveis regulações. A junção do conhecimento biológico e do conhecimento inferido pelos algoritmos,

permite validar mais eficientemente novas regulações, sendo sempre necessária a confirmação laboratorial.

A partir de um genoma sequenciado, é possível obter as regiões promotoras dos genes. Por outro lado, uma vez descoberto por via laboratorial que um factor de transcrição se liga a um determinado *consensus*, é possível, usando algoritmos de emparelhamento de cadeias de caracteres, determinar quais os genes que contêm essa zona de *consensus* na região promotora. Com esta metodologia obtêm-se genes que são potencialmente regulados por esse factor de transcrição. Enquanto não houver verificação laboratorial, estas regulações são identificadas por regulações potenciais.

Capítulo 3

Estruturas de dados

Neste capítulo descreve-se o procedimento seguido para a identificação dos conceitos relevantes e o mapeamento destes conceitos para a definição da estrutura da base de dados utilizada.

A identificação destes conceitos foi conseguida partindo dos conceitos biológicos apresentados no capítulo anterior, e promovendo reuniões periódicas com o grupo de Ciências Biológicas do Departamento de Eng. Química do IST, de forma a tentar perceber e estruturar o conhecimento da área da Biologia Molecular necessário para a construção deste sistema.

Apesar da informação representada ser relativa ao organismo *Saccharomyces cerevisiae*, que é um eucariota simples, este sistema modela adequadamente a maioria das relações existentes nos eucariotas superiores. A maior diferença reside no facto da informação contida num gene, em eucariotas simples, codificar apenas uma proteína, enquanto que em eucariotas superiores um gene poder dar origem a mais do que uma proteína, devido ao *splicing alternativo*.

O modelo utilizado para representar a informação foi o modelo Entidade-Relação (ER). Este modelo permite representar informação do mundo real em termos de conceitos e as suas relações, sendo usado no desenho inicial de bases de dados.

3.1 Identificação dos Conceitos

A informação a representar no sistema, não se encontrava inicialmente estruturada numa base de dados sendo manipulada através de folhas de cálculo, existindo uma folha para cada visão possível do problema. Assim, para encontrar uma característica de um gene a partir do nome

de uma ORF, era necessário ir procurar numa folha de cálculo qual a correspondência entre ORF e genes, e noutra folha de cálculo a característica desejada.

O facto de cada investigador ter na sua posse várias folhas de cálculo traz inúmeras desvantagens, uma vez que sempre que alguém acrescenta informação a uma folha, tem que distribuir por todos os investigadores dessa área. A confusão aumenta no caso de esta divulgação acontecer paralelamente com alterações por parte de outras pessoas. Nesta situação, torna-se útil existir um repositório central actualizado que evite versões concorrentes e que disponha de uma visão integrada dos conceitos, permitindo obter a informação desejada rapidamente.

Após várias reuniões foram identificados três dos conceitos principais. Estes são: o conceito de *ORF/gene*, de Proteína[1] e de *Consensus*. Estes conceitos e a maneira como se relacionam modelam a maioria dos mecanismos de regulação de genes.

3.1.1 Conceito de ORF/Gene

Este conceito partilha dois sub-conceitos, o de ORF e o de gene. Como foi referido anteriormente uma ORF encontra-se flanqueada por um codão de iniciação e um codão de finalização. No entanto, pode não conter uma zona codificante. Assim, todos os genes têm uma ORF associada, mas o contrário pode não ser verdade, o que significa que, nem todas as ORF contêm um gene.

Atributos

Após estabelecer que uma ORF codifica um gene, é atríbuido um nome ao gene consoante a função deste. Para além deste nome podem ainda existir vários nomes alternativos dados por diferentes grupos de investigação.

Todos os genes têm ainda associados a sequência de nucleótidos que codifica a proteína e a sequência de nucleótidos da região promotora onde se ligam os factores de transcrição.

Para além destas características essenciais foram posteriormente adicionadas outras duas: um *link* para a base de dados SGD onde são apresentados dados dessa ORF; e um campo indicando se um determinado gene é um retro-transposão (ver glossário).

Assim, foi criado o conceito de *ORF/gene*, contendo o nome da ORF (**orfname**), o nome do gene (**genename**), os nomes alternativos do gene (**alternativename**), a sequência codi-

[1] Em Inglês, *Protein*.

ficante (**genesequence**), a sequência promotora (**promotersequence**), o *link* para a SGD (**url**) e um campo que indica se é um retro-transposão (**retrotransposon**), como se pode ver na Figura 3.1.

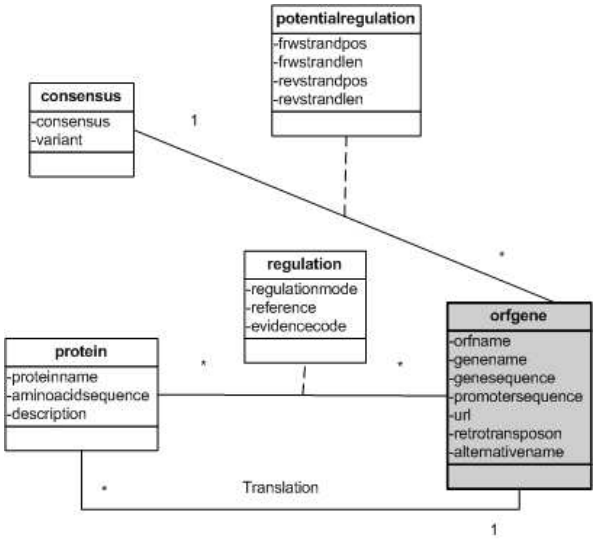


Figura 3.1: Conceito de ORF/gene.

Relações

Este conceito tem associadas três relações: *Translation*, *Regulation* e *PotentialRegulation*. A relação *Translation* é uma relação com o conceito *Protein*. Esta relação representa o facto de a sequência de um gene poder codificar uma ou mais proteínas, e permitir identificar a partir do nome do gene o nome das proteínas correspondentes e vice-versa. No caso do organismo considerado isto não acontece, codificando cada gene apenas uma proteína, mas pode acontecer em organismos eucariotas mais complexos, como por exemplo, o Homem.

Devido ao sentido das relações *Regulation* e *PotentialRegulation*, estas serão explicadas no conceito *Protein* e no conceito *consensus*, respectivamente.

3.1.2 Conceito de *Protein*

As proteínas, como foi explicado anteriormente, são as unidades funcionais da célula. Algumas podem ter a função de factores de transcrição, ligando-se à região promotora de outros genes, sendo responsáveis pela sua transcrição.

Atributos

Neste sistema uma proteína é descrita pelos seguintes atributos: um identificador (**proteinname**), uma sequência de aminoácidos (**aminoacidsequence**) e uma descrição (**description**), como se pode ver na Figura 3.2.

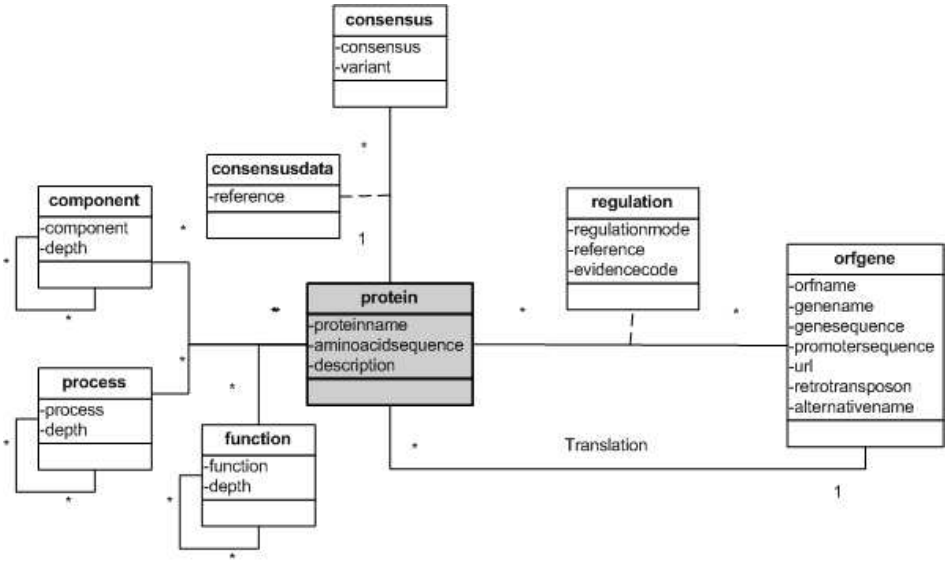


Figura 3.2: Conceito de proteína.

Relações

Apesar de não fazer parte dos atributos de uma proteína existe uma relação, denominada de *consensusdata* entre o conceito *protein* e o conceito *consensus*. Cada proteína, que seja factor de transcrição, pode estar associada a várias zonas de *consensus*. Associada a esta relação entre proteína e zona de *consensus* existe um campo contendo a referência bibliográfica desta relação.

Outros dois tipos de relações interligam o conceito *Protein* com o conceito *ORF/gene*. A primeira relação já foi descrita na secção 3.1.1. A segunda é uma relação de regulação, identificada por *regulation*, entre um factor de transcrição e um ou mais genes. A esta relação está associada o tipo de regulação, activação ou repressão (**regulationmode**), o modo como esta regulação foi obtida (**evidencecode**), e a referência bibliográfica relativa a essa regulação (**reference**).

Encontram-se ainda relacionadas três ontologias do *Gene Ontology Consortium*, indicando a componente celular onde a proteína actua (localização), o processo biológico em que está envolvida (actividade) e a sua função molecular (o seu papel nessa actividade). Estas três ontologias serão apresentadas na secção 3.2.

3.1.3 Conceito de *Consensus*

O conceito *Consensus* está associado a um factor de transcrição, ou seja, a uma proteína que está envolvida na regulação de genes.

Atributos

Este conceito existe com o único objectivo de representar os locais de ligação, zonas de *consensus*, reconhecidas por cada factor de transcrição. Assim, o conceito *Consensus* tem como atributo a sequência de nucleótidos reconhecida por cada factor de transcrição (**consensus**). No caso de existirem vários *consensus* reconhecidos por um determinado factor de transcrição, existe um campo adicional (**variant**), contendo uma letra para os distinguir. Para cada factor de transcrição, este atributo vai sendo incrementalmente associado (A, B, ... Z) a cada zona de *consensus*.

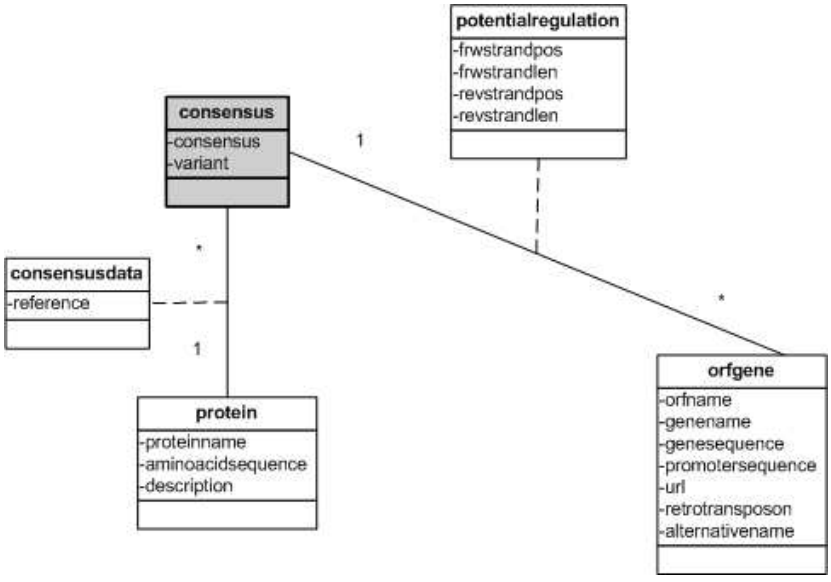


Figura 3.3: Conceito de Consensus.

Relações

É comum um factor de transcrição reconhecer mais do que uma zona de *consensus*, isto é, reconhecer mais do que uma única sequência de nucleótidos. Este facto é representado através de uma relação entre o conceito *Protein* e o conceito *Consensus*, numa relação de ordem de *um-para-muitos*. Cada uma destas relações tem associada a referência bibliográfica para o artigo onde foi descoberta a sequência de nucleótidos indentificada como zona de *consensus*.

Existe ainda outra relação entre o conceito *Consensus* e o conceito *ORF/gene*, devido às regulações potenciais de genes, como explicado na secção 2.5.1. Assim, de cada vez que é inserido um *consensus* no sistema é executado um algoritmo de forma a identificar em que sequências promotoras, em que posição (**frwstrandpos**), qual o seu comprimento (**frwstrandlen**) e em que sentido (**revstrandpos** e **revstrandlen**), a zona de *consensus* aparece. Esta informação fica associada à relação entre um determinado *Consensus* e um determinado *ORF/gene*, Figura 3.3.

3.2 Gene Ontology Consortium

Depois de vários genomas terem sido sequenciados, muita informação necessitava de ser interpretada. Assim, tornou-se necessário atribuir terminologias que, de uma forma sintética e normalizada, representassem as funções biológicas e os processos moleculares em que as proteínas e os genes estão envolvidos. Estas terminologias começaram por ser dependentes do organismo em questão, sendo cada grupo de investigação responsável pela terminologia atribuída ao organismo estudado. Este facto fez proliferar o número de termos utilizados. Outro factor que fez aumentar a diversidade de terminologias foi o facto de existirem processos biológicos num organismo que são inexistentes noutro.

Por forma a contornar esta situação, em 1998 três grupos de investigação, cada um especializado no estudo do um organismo, uniram-se formando o *Gene Ontology Consortium* [5]. O principal objectivo desta colaboração foi arranjar uma terminologia comum para as funções e processos biológicos. Os grupos em questão foram a comunidade de estudo da *Drosophila* (*Flybase* [8]), a comunidade de estudo da *Saccharomyces cerevisiae* (*SGD* [3]) e a comunidade de estudo do rato (*MGI* [9]). Apesar deste esforço de uniformização ter sido iniciado por estas três instituições, foi posteriormente expandido para outras. Actualmente existem dezasseis

instituições envolvidas, sendo que algumas delas estudam mais do que um organismo.

O resultado final deste esforço foram três ontologias[2], uma contendo informação sobre a localização onde a proteína actua (componente celular), outra com informação sobre a actividade em que a proteína está envolvida (processo biológico) e a última indicando o papel específico da proteína nesse processo (função molecular).

Estas três ontologias estão organizadas hierarquicamente através de relações PART_OF (composição) e IS_A (herança), e encontram-se disponíveis em vários formatos [10] (MySQL, XML e outros formatos de texto).

Esta uniformização de termos facilitou a comparação de proteínas de diferentes organismos, facilitando a observação das evoluções entre espécies. A criação destas ontologias veio ainda evitar que diferentes pessoas inserissem diferentes termos com o mesmo significado, permitindo a um computador ter a colecção dos termos existentes, facilitando as pesquisas e a navegação dentro dessas ontologias.

Cada uma destas ontologias está organizada num grafo, existindo um termo raíz, que corresponde ao topo da hierarquia e em que os restantes termos podem ter mais do que um termo pai.

Nesta secção são descritas as três ontologias do *Gene Ontology Consortium* e a forma como estas foram representadas no sistema desenvolvido.

3.2.1 Function

Este conceito representa a função molecular de uma proteína. Como foi referido anteriormente, cada uma destas ontologias relaciona os termos numa hierarquia de forma a poder identificar sub-funções (relações IS_A) ou várias funções que compõem uma outra (relações PART_OF).

Atributos

Visto que os conceitos estão organizados numa hierarquia, existe um atributo para indicar a que distância se encontra um determinado conceito do nó raíz, denotando a sua profundidade. Existe também um atributo com a descrição da função molecular.

[2]Uma ontologia é definida por um conjunto de conceitos, em que se chegou a um consenso na definição de cada um, e nas suas relações.

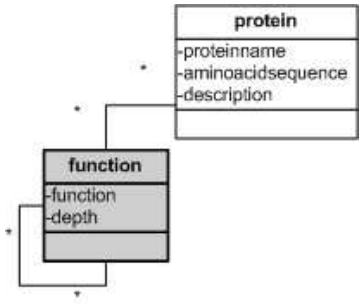


Figura 3.4: Conceito de função molecular.

Relações

Cada proteína pode ter associadas várias funções moleculares, dando origem a uma relação de *um-para-muitos* entre proteína e função molecular. Por outro lado, como cada função molecular pode ter várias proteínas associadas, essa relação é na verdade de *muitos-para-muitos*.

3.2.2 Process e Component

O conceito *Process* representa os termos do *Gene Ontology Consortium* relativos ao processo biológico em que uma determinada proteína se encontra envolvida, e o conceito *Component* representa os termos relativos à componente celular em que uma determinada proteína se encontra.

Estes dois conceitos representam informação distinta do conceito *function*. No entanto a forma como a informação representada está organizada é semelhante, na medida em que também é utilizada uma hierarquia de termos, em que cada nó contém os mesmos atributos, relacionando-se da mesma forma com o conceito *Protein*.

3.3 Modelo Conceptual

O estado actual do modelo conceptual é apresentado na Figura 3.5. São representados os conceitos *ORF/gene*, *Protein* e *Consensus* e os conceitos referentes às três ontologias do *Gene Ontology Consortium*. Estes conceitos são essenciais para a inferência das redes de regulação de genes.

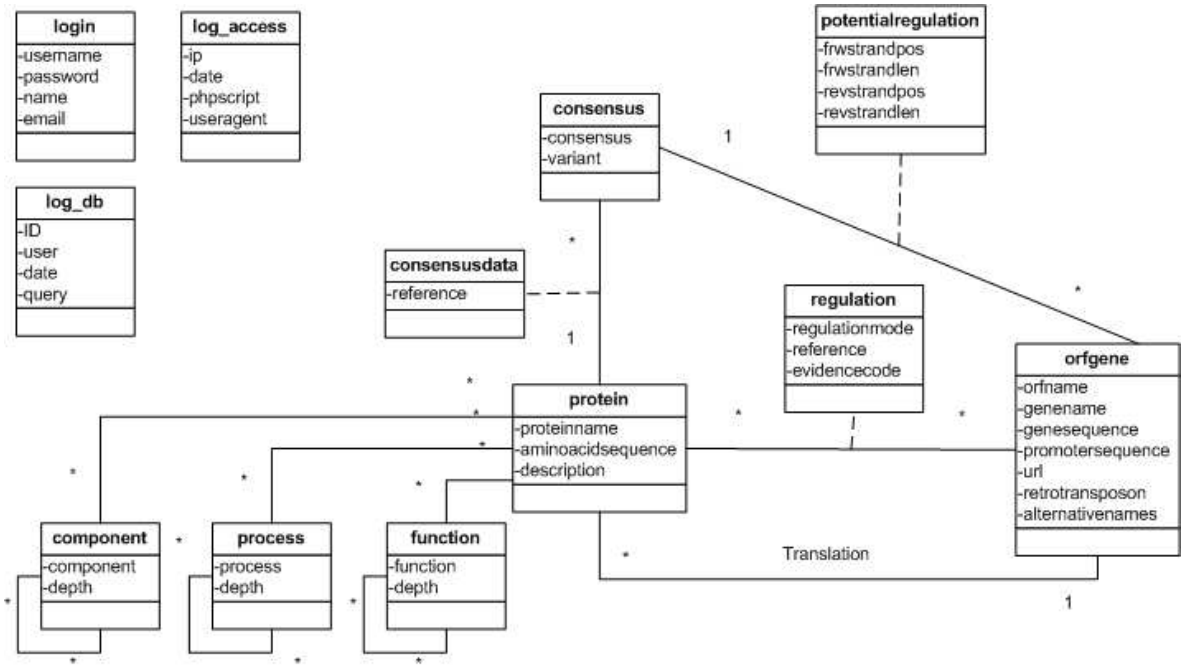


Figura 3.5: Modelo conceptual da base de dados.

Ao longo do desenvolvimento deste processo, este diagrama foi evoluindo progressivamente sempre que existiu a necessidade de modelar fenómenos mais complexos.

O conceito *ORF/gene* estava inicialmente separado em ORF e gene, embora se tivesse rapidamente apercebido que representavam a mesma informação.

A relação *regulation* que actualmente representa as relações documentadas de activação e de repressão, inicialmente representava apenas as relações documentadas de activação, tendo sido corrigida através da adição do atributo **regulationmode**.

Inicialmente, o conceito de *consensus* não tinha o seu papel bem definido, no que diz respeito à sua relação com os factores de transcrição. Actualmente, a relação deste conceito com o conceito de proteína, permite distinguir as proteínas que têm o papel de factores de transcrição das que não têm. O aparecimento do conceito de *consensus*, permitiu ainda a definição de uma relação que representasse as regulações potenciais entre este conceito e o conceito de *ORF/gene*.

Existiu ainda a necessidade de representar informação relativa ao acesso dos utilizadores, tanto dos curadores através do *backoffice* (ver secção 4.4.4), como do resto dos utilizadores através da página pública, de forma a efectuar estatísticas de utilização.

Capítulo 4

Sistema de informação

No capítulo anterior foram identificados os conceitos essenciais para a representação das redes de regulação e a forma como estavam relacionados. Neste capítulo será apresentada a arquitectura do sistema, dando ênfase à forma como é feita a comunicação com a base de dados. São ainda apresentados vários procedimentos de extracção, tratamento e carregamento dos dados na base de dados.

4.1 Arquitectura

O sistema foi desenvolvido em PHP e é suportado por um servidor *Web* com um módulo de PHP incluído, comunicando com um sistema de gestão de base de dados (SGBD). Para o SGBD foi escolhido o *MySQL*.

A arquitectura deste sistema apresenta a típica divisão em três camadas, correspondendo à separação entre base de dados, serviços e interface com o utilizador, descrito na Figura 4.1.

Tanto os utilizadores de acesso público como de acesso restrito, acedem a estes serviços através de um *browser*. Estes serviços encontram-se divididos em dois grandes grupos. Os serviços de *frontoffice* que incluem todas as funcionalidades disponíveis ao utilizador comum, e os serviços de *backoffice* que incluem a inserção de novos dados e manutenção dos existentes. No capítulo 5, serão apresentados os problemas existentes, e a abordagem efectuada para os solucionar.

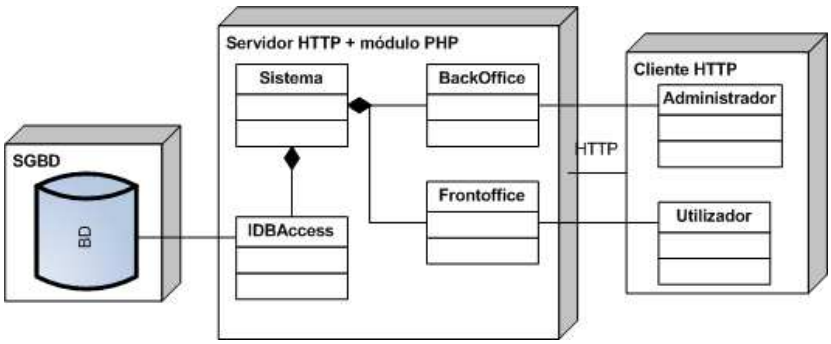


Figura 4.1: Arquitectura do sistema de informação.

4.1.1 Escolhas de implementação

Serão descritas de seguida algumas das opções de implementação consideradas.

O mecanismo de armazenamento e indexação utilizado na base de dados foi o *MyISAM*. Este mecanismo utiliza uma estrutura de dados em árvore, denominada *B-tree*. Esta estrutura tem a característica de se encontrar sempre equilibrada, o que faz com que a profundidade das várias folhas da árvore não difira mais de uma unidade. Os dados neste mecanismo de armazenamento e indexação encontram-se ordenados sequencialmente, não tendo muitas páginas de *overflow* em cada folha, tornando-o óptimo para pesquisas sequenciais. No caso da árvore sofrer muitas alterações, irão aparecer páginas de *overflow* nas folhas da árvore, tornando a pesquisa sequencial menos eficiente. Como o sistema desenvolvido não está sujeito a muitas alterações, mas sim pesquisas, a escolha do mecanismo *MyISAM* revelou-se bastante relevante.

As chaves primárias das tabelas vão corresponder ao índice principal pelo qual o mecanismo *MyISAM* vai indexar os dados.

Foi usado o tipo VARCHAR para os identificadores de quase todas as tabelas. Este tipo é em tudo semelhante ao tipo CHAR, com a diferença de o número de caracteres corresponder ao número máximo de caracteres representado por esse campo. Este tipo de dados permite assim poupar espaço nas tabelas da base de dados.

O tipo de dados TEXT foi usado para representar sequências de nucleótidos ou aminoácidos. Os campos com este tipo de dados não contêm nenhum índice associado. Por outro lado, para representar os índices das tabelas foi sempre usado o tipo de dados VARCHAR, para representar dados alfa-numéricos, e o tipo de dados INT para representar dados inteiros.

4.2 Modelo físico

Nesta secção será descrito o modelo físico do sistema desenvolvido. Convém, notar que a passagem do modelo conceptual para o modelo físico é dependente do sistema de gestão de base de dados utilizado. Muitos destes sistemas, apesar de conterem os mesmos tipos básicos de informação, apresentam ligeiras diferenças relativamente a tipos de dados mais complexos.

No modelo físico, os conceitos e as suas relações dão origem a tabelas e os atributos dão origem a colunas na base de dados. Estas tabelas e colunas vão conter várias restrições, tais como chaves primárias, chaves estrangeiras ou chaves únicas.

Em função das relações entre os conceitos do modelo conceptual podem, no modelo físico, surgir tabelas auxiliares para representar essas relações. O modelo físico é apresentado na Figura 4.2. As tabelas *translation*, *regulation*, *processlist*, *componentlist*, *functionlist*, *consensusdata* e *potentialregulation* representam relações presentes no modelo conceptual apresentado na Figura 3.5.

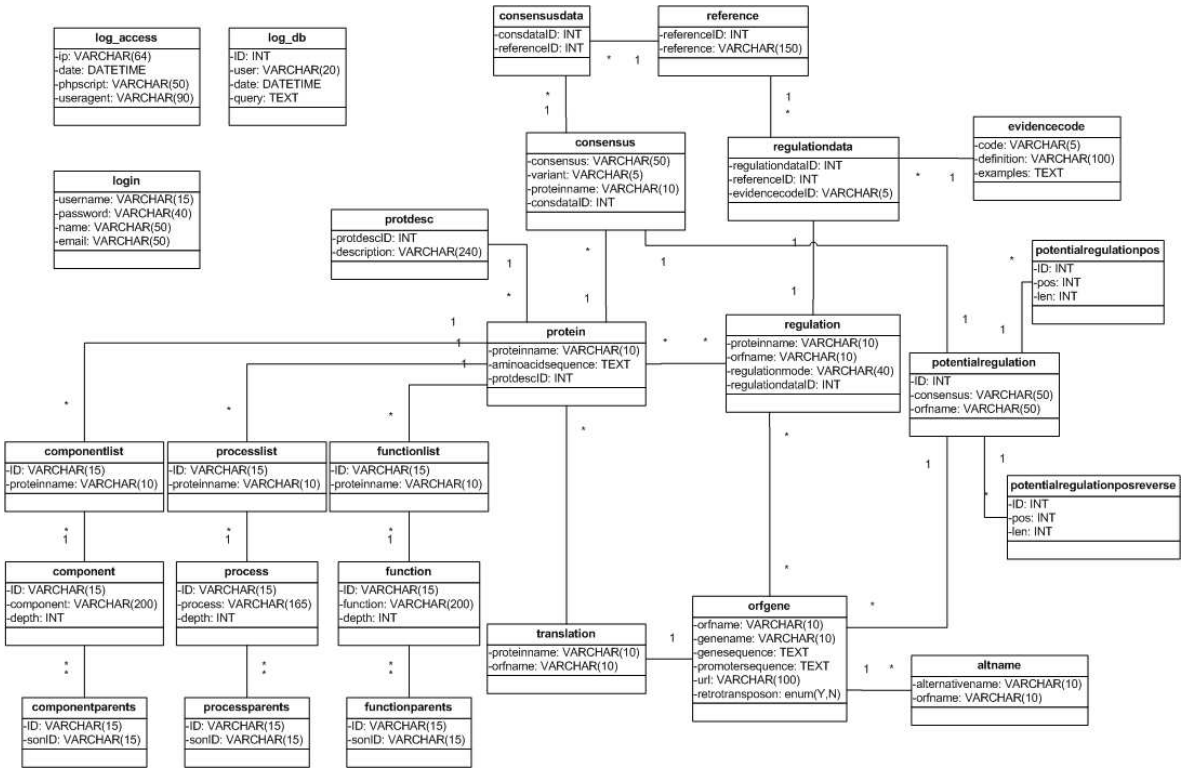


Figura 4.2: Modelo físico da base de dados.

4.2.1 Tabelas relacionadas com o conceito *ORF/gene*

Tabela *orfgene*

No modelo físico existe uma tabela correspondente ao conceito *orfgene* do modelo conceptual. A Figura 4.3 representa essa tabela.

Figura 4.3: Tabela *orfgene*.

Praticamente todos os atributos do conceito estão representados nas colunas dessa tabela, sendo estas identificadas pelo mesmo nome existente no conceito. A excepção é o atributo correspondente ao nome alternativo do gene que não aparece nesta tabela do modelo físico. Este facto será explicado aquando da descrição da tabela seguinte, a tabela *altname*.

Visto que o atributo **orfname** existe sempre e o atributo **genename** pode não existir, o atributo **orfname** foi considerado como sendo chave primária, isto é, identificador único, da tabela *orfgene*. Esta tabela encontra-se descrita pelo Código SQL 8.1 no apêndice 8.5.

Os campos **orfname** e **genename** para o organismo em estudo têm identificadores pequenos, sete caracteres no máximo. No entanto, na salvaguarda de algum caso especial, foram definidos como podendo ter no máximo até dez caracteres, sendo do tipo VARCHAR(10).

Em termos de índices, para além do índice da chave primária, foi criado um segundo índice no campo **genename** para acelerar as pesquisas pelo nome do gene.

Tabela *altname*

A tabela auxiliar *altname*, Figura 4.4, surge devido à possibilidade de existir em vários nomes de gene, nomes alternativos para a mesma ORF. Esta tabela contém os nomes alternativos de todos os genes, tendo dois campos: o nome alternativo de determinado gene (**alternativename**) e o nome da ORF a que está associado (**orfname**), conforme ilustrado

no Código SQL 8.2 no apêndice 8.5.

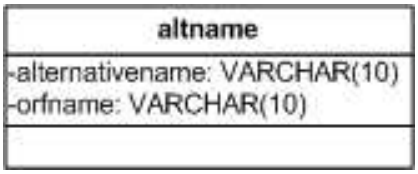


Figura 4.4: Tabela *altname*.

Os tipos de dados usados nos campos desta tabela foram os mesmos dos campos **orfname** e **genename** da tabela *orfgene*, visto guardarem o mesmo tipo de informação.

Em termos de índices, temos como índice primário dois campos: o **orfname** e o **alternativename**. No entanto, para acelerar as pesquisas pelo nome alternativo de um gene, foi criado um índice adicional no campo **alternativename**.

Tabela *translation*

Para representar a relação *translation* definida entre o conceito *orfgene* e o conceito *protein*, foi criada uma tabela na base de dados com o mesmo nome. A Figura 4.5 mostra esta tabela. No anexo 8.5 esta tabela está descrita pelo Código SQL 8.3.

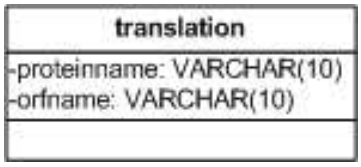


Figura 4.5: Tabela *translation*.

Face à relação de *um-para-um*, apenas neste organismo, entre o conceito *orfgene* e o conceito *protein*, esta tabela de relação não é estritamente necessária. Em alternativa poderia ser incluído na tabela *protein* uma chave estrangeira com o campo **orfname**. A razão da existência desta tabela prende-se com uma possível evolução do sistema para outros organismos mais complexos e também com a optimização dos acessos à base de dados. Como em muitas das funcionalidades é necessário fazer a tradução entre o nome da ORF e o nome da proteína, torna-se mais eficiente utilizar uma tabela que contém menos colunas e dados.

Esta tabela é composta pelos identificadores da tabela *orfgene* e da tabela *protein*, tendo como única função permitir a tradução do conceito *orfgene* para o conceito *protein* e vice-versa. Tal como o campo **orfname** da tabela *orfgene* e o campo **proteinname** da tabela *protein*, os campos desta tabela são do tipo VARCHAR(10) uma vez que guardam o mesmo tipo de informação.

Em termos de índices, o índice principal é obtido através do campo **orfname** e do campo **proteinname**. Para acelerar as pesquisas pelo nome da proteína, foi criado um índice adicional no campo **proteinname**.

4.2.2 Tabelas relacionadas com o conceito *Protein*

Tabela *protein*

Esta tabela contém os mesmos atributos do conceito que lhe deu origem. A Figura 4.6 representa esta tabela. No entanto, existe uma ligeira diferença, o campo **description** do conceito foi substituído pelo campo **protdescID** referenciando a tabela *protdesc*.

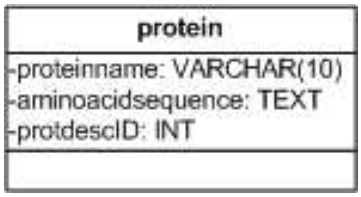


Figura 4.6: Tabela *protein*.

A tabela *protein* contém também um identificador denominado **proteinname**, do tipo VARCHAR(10). O campo contendo a sequência dos aminoácidos tem como nome **aminoacidsequence**, sendo do tipo TEXT. A escolha deste tipo de dados prende-se com o facto da sequência de aminoácidos poder variar bastante entre proteínas. A definição da tabela encontra-se descrita no Código SQL 8.4 no anexo 8.5.

Em termos de índices o identificador da proteína, **proteinname**, é a chave primária da tabela, e será referenciado por outras tabelas como chave estrangeira. O campo **protdescID** é chave estrangeira para a tabela *protdesc*, representando a descrição da proteína.

Tabela *protdesc*

A tabela *protdesc*, ilustrada na Figura 4.7, foi criada para conter a descrição de uma proteína e encontra-se documentada no anexo 8.5 no Código SQL 8.5.

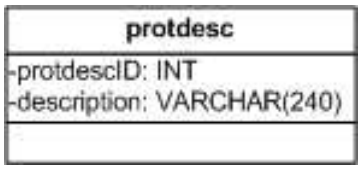


Figura 4.7: Tabela *protdesc*.

A escolha de criar uma tabela auxiliar em vez de guardar o campo **description** na própria tabela *protein* teve como objectivo evitar duplicações e manter a normalização da base de dados, visto que por vezes existe a mesma descrição partilhada por várias proteínas. Assim, cada descrição é guardada uma única vez nesta tabela e na tabela *protein* existe uma referência para essa descrição.

Esta tabela é composta pelo campo **description**, contendo a descrição das proteínas, sendo do tipo VARCHAR(240), visto que não existem descrições maiores do que 240 caracteres. A escolha do tipo de dados VARCHAR em vez do tipo de dados TEXT tem a ver com o facto do tipo VARCHAR ter associada uma chave única, sendo assim mais eficiente. O outro campo é o **protdescID** que é o identificador único da tabela.

Em temos de índices, o campo **protdescID** é a chave primária da tabela visto que é chave estrangeira na tabela *protein*, e o campo **description** tem uma chave única visto que não queremos descrições duplicadas na tabela.

Tabela *regulation*

Para representar a relação *regulation* definida anteriormente, entre o conceito *protein* e o conceito *orfgene*, foi criada uma tabela na base de dados com o mesmo nome, conforme ilustrada na Figura 4.8. A definição da tabela encontra-se descrita no anexo 8.5 no Código SQL 8.6.

Esta tabela tem como colunas os identificadores da tabela *protein* e da tabela *orfgene*. Os tipos de dados usados nestas colunas foram os mesmos do campo **proteinname** da tabela *protein* e do campo **orfname** da tabela *orfgene*, visto que guardam o mesmo tipo de

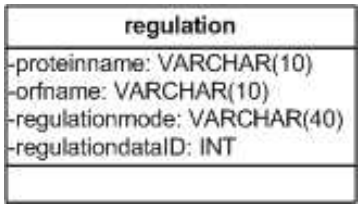


Figura 4.8: Tabela *regulation*.

informação.

O campo **regulationmode** existente no conceito, constituirá uma coluna nesta tabela. Este campo pode tomar quatro valores diferentes, 'NULL', 'activator', 'repressor' e 'activator/repressor', indicando o tipo de regulação entre um factor de transcrição e uma ORF. O valor 'NULL' é utilizado no caso de não ser conhecido o tipo de regulação, o valor 'activator' no caso de ser uma regulação activadora, o valor 'repressor' no caso de ser uma regulação repressora, e o valor 'activator/repressor' no caso da regulação poder ser dos tipos activação e repressão.

Os campos **reference** e **evidencecode** não constituem directamente colunas desta tabela. No entanto estes campos estão contidos numa tabela denominada *regulationdata*, que será definida posteriormente e que contém um identificador único para cada associação entre os dois campos, denominado **regulationdataID**. Esta escolha de implementação será explicada em detalhe na tabela *regulationdata*. Neste momento vamos apenas considerar que este novo identificador contém a mesma informação que os outros dois campos, e que constitui um dos campos da tabela *regulation*.

Tabela *regulationdata*

Esta tabela surge com o objectivo de juntar num só identificador toda a informação relativa a referências e condições experimentais pelas quais se obteve uma determinada regulação, conforme ilustrada na Figura 4.9. A definição desta tabela é apresentada no anexo 8.5 no Código SQL 8.7.

Esta tabela contém o identificador da referência da regulação, **referenceID**, que é chave estrangeira para a tabela *reference*. Este identificador é um inteiro. Contém ainda um identificador das condições experimentais, **evidencecodeID**, que é chave estrangeira da tabela

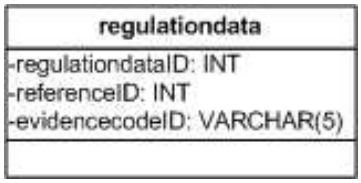


Figura 4.9: Tabela *regulationdata*.

envidencecode, explicada a seguir. Este campo, que contém o acrónimo correspondente à condição experimental, é do tipo VARCHAR(5).

O campo **regulationdataID** é a chave primária desta tabela e representa a junção da informação representada pelos outros dois identificadores.

Tabela *reference*

Esta tabela, descrita na Figura 4.10, tem como objectivo guardar as referências tanto de regulações entre um factor de transcrição e uma ORF, bem como de *consensus* associados a factores de transcrição. A definição desta tabela é apresentada no anexo 8.5 no Código SQL 8.8.

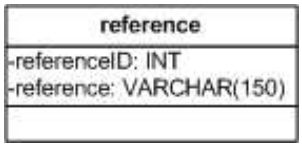


Figura 4.10: Tabela *reference*.

A tabela é composta por dois campos: o campo **reference** que contém a descrição da referência e o campo **referenceID** que contém o identificador a ser referenciado nas tabelas *consensusdata* e *regulationdata*.

Tabela *evidencecode*

Esta tabela contém a informação relativa ao método pelo qual a informação foi obtida, de modo a suportar a validade dos dados. Esta tabela encontra-se descrita na Figura 4.11. A descrição desta tabela é apresentada no anexo 8.5 no Código SQL 8.9.

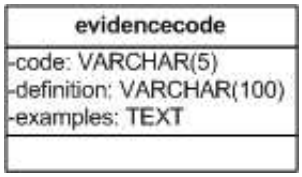


Figura 4.11: Tabela *evidencecode*.

Esta informação é composta por um acrónimo, **code**, uma descrição desse acrónimo, **description**, e alguns exemplos de casos práticos em que determinado código pode ser utilizado, **examples**.

Tabela *functionlist*, *processlist* e *componentlist*

Para representar a relação entre o conceito *protein* e cada um dos conceitos *function*, *process* e *component* foram criadas três tabelas na base de dados. A primeira com o nome de *functionlist*, a segunda com o nome de *processlist* e a última com o nome de *componentlist*. Estas tabelas encontram-se descritas na Figura 4.12. A definição de cada uma delas é apresentada no anexo 8.5 no Código SQL 8.10, 8.11 e 8.12.

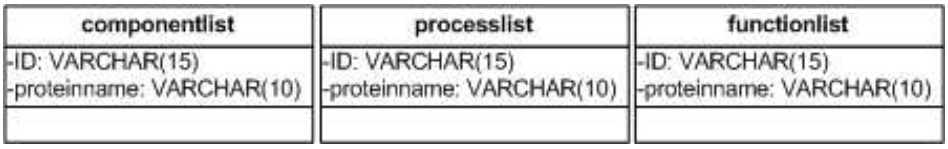


Figura 4.12: Tabelas *functionlist*, *processlist* e *componentlist*.

Cada uma destas tabelas é composta pelo identificador da tabela *protein*, **proteinname** e pelo identificador **ID** da tabela do *Gene Ontology Consortium* correspondente, *function*, *process* ou *component*. Este identificador faz a ligação entre uma determinada proteína e cada uma das tabelas.

Em termos de índices, o índice principal é o obtido através do campo **proteinname** e do campo **ID**. Para acelerar as pesquisas pelo identificador da função, foi criado um índice adicional no campo **ID**.

4.2.3 Tabelas relacionadas com o conceito *Consensus*

Tabela *consensus*

Esta tabela, apresentada na Figura 4.13, contém os atributos do conceito que lhe deu origem: o atributo **consensus** que guarda a sequência de nucleótidos e o atributo **variant** que representa a variante do *consensus*. A definição da tabela é apresentada no anexo 8.5 no Código SQL 8.13.

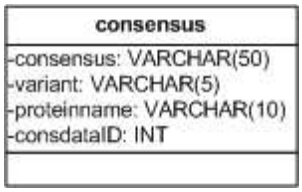


Figura 4.13: Tabela *consensus*.

Contém ainda dois atributos adicionais: o **proteinname** que é a chave estrangeira para a tabela *protein*; e o **consdataID** que é chave estrangeira para a tabela *consensusdata*. O aparecimento da tabela auxiliar *consensusdata* será explicado a seguir.

Tabela *consensusdata*

Esta tabela, apresentada na Figura 4.14, surge com o objectivo de juntar num só identificador toda a informação relativa a referências bibliográficas e condições experimentais pelas quais se obteve um determinado **consensus**. A definição da tabela é apresentada no anexo 8.5 no Código SQL 8.14.

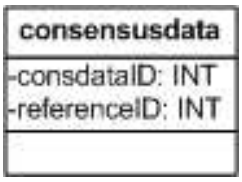


Figura 4.14: Tabela *consensusdata*.

O identificador **consdataID**, é posteriormente referenciado como chave estrangeira na tabela *consensus*, representando toda a informação. No entanto, como ainda não existem dados para as condições experimentais, a tabela apenas contém o identificador para a tabela

reference. No caso de existirem dados para preencher as condições experimentais, basta acrescentar um identificador a esta tabela. Um caso em que esta junção de informação já acontece é na tabela *regulationdata* em que se junta a referência e as condições experimentais de uma regulação.

Em termos de índices, o campo **consdataID** é a chave primária da tabela, e o campo **referenceID** é a chave estrangeira para a tabela *reference*. Todos os campos existentes nesta tabela são inteiros, devido a conterem apenas um identificador numérico.

Tabela *potentialregulation*

Esta tabela representa a relação existente entre o conceito *consensus* e o conceito *orfgene*, no modelo conceptual. Esta tabela representa o facto de determinada sequência de nucleótidos, representada no campo **consensus**, existir na região promotora de uma ORF, representada pelo seu nome **orfname**. Esta tabela encontra-se apresentada na Figura 4.15. A definição da tabela é apresentada no anexo 8.5 no Código SQL 8.15.

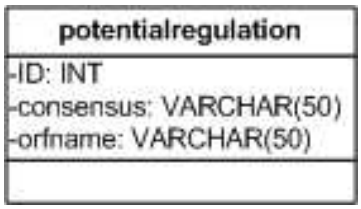


Figura 4.15: Tabela *potentialregulation*.

Visto essa sequência de nucleótidos se poder ligar em qualquer dos sentidos da cadeia de ADN, foram criadas duas tabelas auxiliares, explicadas de seguida. A tabela *potentialregulationpos* que contém as posições na região promotora do gene em que a sequência de nucleótidos foi encontrada, quando esta é percorrida no sentido directo. E a tabela *potentialregulationposreverse* que contém a mesma informação mas quando a região promotora do gene é percorrida no sentido inverso.

Os campos **consensus** e **orfname** vão constituir a chave primária desta tabela, visto que representam a relação entre os dois conceitos. Foi criado ainda um campo denominado **ID**, para ser referenciado nas duas tabelas auxiliares. A este identificador está associada uma chave única de forma a que não existam dois identificadores iguais.

Tabela *potentialregulationpos* e *potentialregulationposreverse*

Estas duas tabelas auxiliares contêm as posições em que a sequência de *consensus* aparece na região promotora do gene, uma lida no sentido directo, e outra lida no sentido inverso. Ambas as tabelas são apresentadas na Figura 4.16. A definição das tabelas é apresentada no anexo 8.5 no Código SQL 8.16 e 8.17.

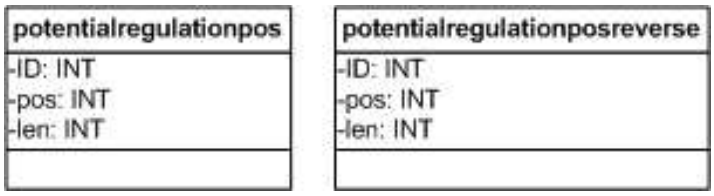


Figura 4.16: Tabelas *potentialregulationpos* e *potentialregulationposreverse*.

O campo **ID** é o identificador da relação entre um *consensus* e uma ORF. Associado a este identificador está a posição, **pos**, visto que um *consensus* pode aparecer mais do que uma vez na mesma região promotora, desde que em diferentes posições. Estes dois campos constituem a chave primária desta tabela. Existe ainda um campo adicional, **len**, que indica o tamanho do **consensus** encontrado na região promotora do gene.

4.2.4 Tabelas relacionadas com o *Gene Ontology Consortium*

As tabelas descritas de seguida foram criadas para representar as três hierarquias do *Gene Ontology Consortium*. Para cada uma das hierarquias foram criadas duas tabelas. Uma que contém as terminologias e outra para modelar as relações de parentesco entre terminologias.

Tabela *function*, *process* e *component*

Cada uma destas tabelas representa os nós das hierarquias do *Gene Ontology Consortium*. Estas tabelas encontram-se apresentadas na Figura 4.17. A definição das tabelas é apresentada no anexo 8.5 no Código SQL 8.18, 8.19 e 8.20.

Cada nó é definido por um identificador **ID**, que é único para os todos os termos existentes. Um exemplo do formato deste identificador é o seguinte: "GO:0035170".

Associado ao identificador, encontra-se a descrição do termo, **function**, **process** ou **component**, consoante esse nó pertencer à hierarquia da função molecular, do processo biológico

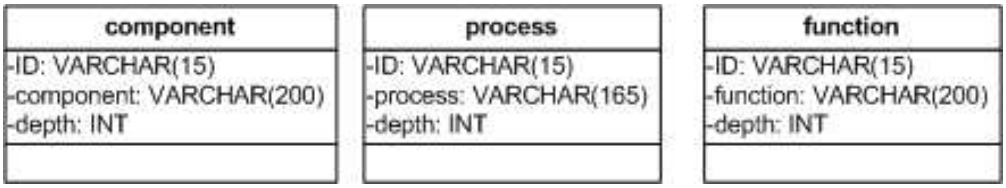


Figura 4.17: Tabelas *function*, *process* e *component*.

ou da componente celular.

Existe ainda um campo a indicar a profundidade do nó na hierarquia correspondente, **depth**. Esta informação é necessária para algumas das funcionalidades desenvolvidas na base de dados.

Tabela *functionparents*, *processparents* e *componentparents*

Cada uma das tabelas seguintes representa as ligações de parentesco entre os nós das três hierarquias. As tabelas encontram-se apresentadas na Figura 4.18. A definição das tabelas é apresentada no anexo 8.5 no Código SQL 8.21, 8.22 e 8.23.

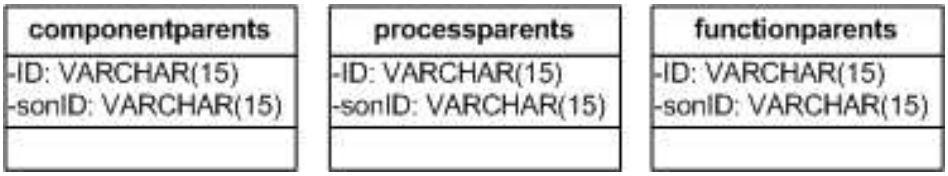


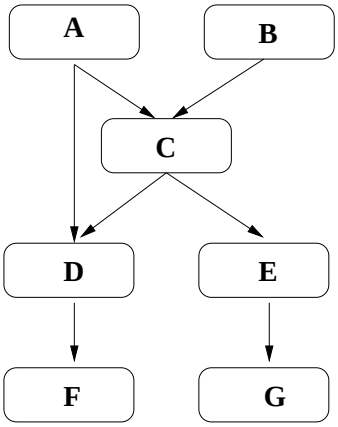
Figura 4.18: Tabelas *functionparents*, *processparents* e *componentparents*.

Como estas tabelas são tabelas de relação, vão apenas conter dois campos, o do nó pai **ID** e o do nó filho **sonID**, como podemos observar na Figura 4.19.

Esta relação é efectuada através de uma chave estrangeira associada de cada campo à tabela da hierarquia correspondente. Os dados que constam nestes campos, são os identificadores únicos do *Gene Ontology Consortium*, "GO:xxxxxx".

4.3 Acesso à base de dados

Sendo este um trabalho académico, o *software* utilizado foi escolhido dentro do leque existente do *software* livre. O motor de base de dados escolhido foi o *MySQL*, devido à simplicidade de



Function

ID	Function
1	A
2	B
3	C
4	D
5	E
6	F
7	G

FunctionParents

ID	sonID
1	3
2	3
1	4
3	4
3	5
4	6
5	7

Figura 4.19: Relação entre a hierarquia de termos e as tabelas da base de dados (exemplo para o conceito *function*).

instalação e devido também à rapidez de resposta às *queries* efectuadas. A versão actualmente usada do *MySQL* é a 4.0.18-Max-log.

Uma característica que influencia a rapidez do *MySQL* é o facto de não fazer verificações de chaves estrangeiras[1]. Esta característica pode ser considerada como desfavorável para o *MySQL*. No entanto, ao ser o próprio programador a fazer as verificações de inconsistência da base de dados, este fica com o controlo das verificações podendo efectuar apenas as *queries* essenciais, tendo ainda a possibilidade de apresentar as mensagens de erro apropriadas.

4.3.1 Camada de abstracção de acesso à base de dados

A linguagem de programação em que este trabalho foi desenvolvido foi o *PHP*, que é uma linguagem de *scripting*, vocacionada para a *Web*. Não é uma linguagem recomendada para projectos que envolvam muitos recursos humanos ou uma boa estruturação de código, visto não existir uma clara separação entre o código de acesso à base de dados, o dos serviços e o da apresentação. Isto significa que muito do código que executa os serviços está junto com o código de apresentação e com o *HTML*.

Assim, devido a não existir uma separação natural entre o código, esta separação fica inteiramente ao critério do programador. Inicialmente, o código responsável pelo acesso à base de dados estava juntamente com o código dos serviços, existindo assim, chamadas específicas

[1]Em inglês, *Foreign Keys*.

de funções de *MySQL* no meio do código.

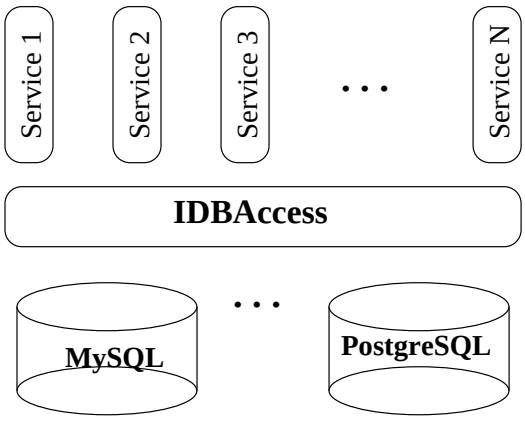


Figura 4.20: Camada de acesso à base de dados.

A portabilidade do sistema de gestão de base de dados era assim mais difícil, sendo necessário verificar a existência de funções específicas em todo o código. Para resolver este problema foi desenvolvida, em *PHP*, uma classe de acesso à base de dados (ver apêndice 8.1), de forma a camuflar a especificidade do acesso de cada base de dados, libertando assim o programador da chamada a funções específicas.

Como se pode ver na Figura 4.20, todos os serviços existentes comunicam com a classe desenvolvida, *IDBAccess*. Assim, os serviços apenas têm o código de comunicação com esta nova classe, e esta contém o código necessário para a comunicação com o SGBD pretendido.

Depois de perceber quais os tipos de comunicação necessários entre a interface e a base de dados, foi desenvolvida a classe genérica *IDBAccess* que providencia métodos abstractos que escondem os métodos específicos fornecidos por classes específicas. Cada classe específica contém as funções específicas de acesso ao SGBD e faz a transformação necessária entre as chamadas ao SGBD e a classe genérica *IDBAccess*.

Aquando da instanciação da classe *IDBAccess* (ver apêndice 8.2), é passado o nome do SGBD pretendido por parâmetro. Assim sempre que se invocar um método desta instanciação, este invocará o método da classe específica indicada aquando da sua criação, e esta classe comunicará com a base de dados em si, fazendo as transformações necessárias à *query* original. Na devolução de resultados, a classe específica é também responsável pela transformação destes para o formato que a classe genérica exige.

4.4 Extracção, Tratamento e Carregamento da Informação

Este sistema de informação teve, tal como a maioria dos sistemas de informação, de ser preenchido com dados. Para tal foi necessário extrair a informação necessária de uma determinada fonte (extracção), fazer o mapeamento dessa informação para a estrutura interna dos dados na base de dados (transformação) e finalmente carregar os dados correctamente validados para a base de dados (carregamento).

Para cada tipo de informação foi implementado um procedimento de extracção, transformação e carregamento.

4.4.1 Lista inicial de genes

Para iniciar a extracção da informação foi necessário obter uma lista com o nome de todas as ORF. O ponto de partida foi o *site* do *RSA tools* [11]. Através deste *site* foi pedida uma lista de todas as ORF existentes em *Saccharomyces cerevisiae*. Esta lista foi recebida por *mail* num ficheiro de texto simples.

4.4.2 Web Spider

Como fonte de informação fidedigna para a extracção da informação foi utilizada a *SGD*, uma vez que contém muitas das informações relativas ao organismo considerado. Todas as informações existentes neste *site* sofreram um processo de verificação e validação por vários investigadores da área da Biologia Molecular, sendo assim bastante fidedignas.

Neste *site* a partir do nome da ORF ou do gene, é possível chegar à restante informação. Todas as informações relativas a cada ORF ou gene encontram-se dispersas por três páginas *Web*. Como a lista de ORF contém 6338 entradas, seria necessário extrair informação de 19014 páginas *Web*.

Para evitar o penoso processo manual, foi desenvolvido um *Web Spider* em *Java* 4.21, utilizando a tecnologia HTTPUnit [12] para a manipulação das páginas HTML.

A biblioteca HTTPUnit foi inicialmente desenvolvida com o objectivo de permitir aos programadores efectuar testes de interface ao seu código de forma automática, permitindo assim testar algo mais do que apenas os serviços.

Neste caso, o HTTPUnit foi usado pela sua capacidade de fazer pedidos HTTP de forma

a extrair informação de várias páginas *Web*, filtrar a informação relevante e guardá-la numa base de dados, sendo tudo isto feito de forma automática.

O HTTPUnit emula parte do comportamento de um *browser*, permitindo a submissão de formulários, suporte de *Javascript*, autenticação HTTP, *cookies* e redireccionamento automático de páginas. Possibilita ainda a extracção de informação de uma página *Web* sob várias formas: sob a forma de texto, em que o programador trata toda a página como uma cadeia de caracteres; sob a forma de tabelas, em que cada tabela corresponde a uma matriz cujas entradas contêm a informação pretendida; sob a forma de uma lista das hiperligações contidas na página; e sob a forma de uma lista de formulários, possibilitando assim a inserção de valores de forma a poder obter a página seguinte.

Para além da biblioteca HTTPUnit, o *Web Spider* desenvolvido utiliza também a biblioteca de acesso à base de dados JDBC [13]. Esta biblioteca contém funções de baixo nível para acessos de leitura e escrita, permitindo a definição de funções mais genéricas para a leitura e escrita dos nossos próprios objectos ou mesmo listas de objectos.

Arquitectura

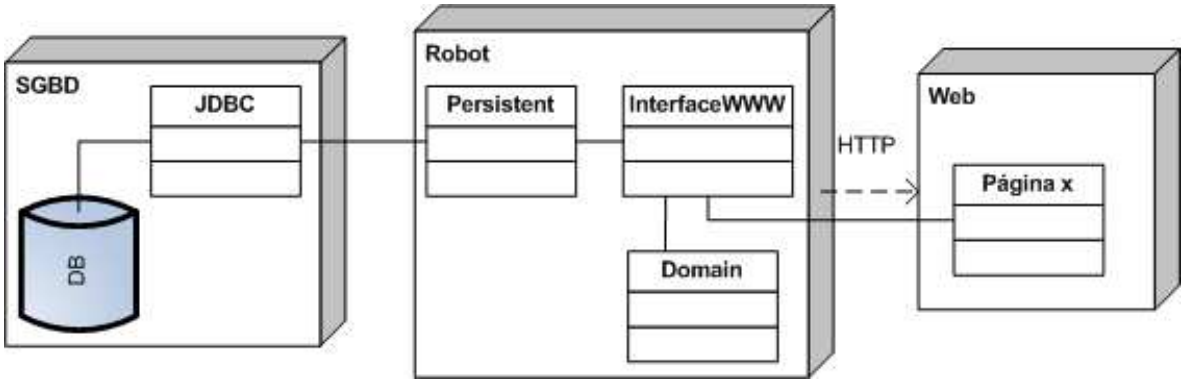


Figura 4.21: Arquitectura do *Web Spider*.

Como podemos ver na Figura 4.22, o *Web Spider* encontra-se dividido em vários pacotes de classes, contendo cada um destes pacotes uma função específica. De seguida são descritos cada um destes pacotes.

- **Config** O pacote *Config* é composto por uma classe que contém as informações de acesso à base de dados: a sua localização, o seu nome, o nome de acesso e a palavra

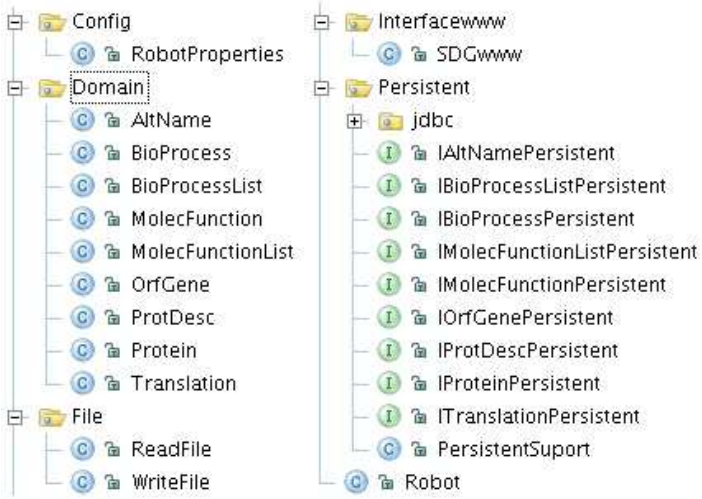


Figura 4.22: Divisão em classes do Web Spider.

chave.

- **Domain** O pacote *Domain* é composto pelos vários objectos a representar. Estes vão corresponder às tabelas da base de dados.

  Estes objectos são instanciados, e após a aquisição da informação necessária na *Web*, são passados ao pacote responsável pela escrita na base de dados.

- **File** O pacote *File* é composto por duas classes. A primeira permite a escrita para um ficheiro e a segunda permite a leitura de dados a partir de um ficheiro.

  Esta classe de escrita tem como objectivo a criação de ficheiros de *log*, registando todos os erros que surgem durante a busca da informação na *Web*.

  A classe de leitura é utilizada para ler a lista de ORF a partir de um ficheiro.

- **Persistent** Este pacote contém as classes responsáveis pela persistência dos objectos existentes no pacote *Domain*. O pacote encontra-se dividido em duas partes. A primeira parte contém apenas a definição da interface de acesso à persistência. A segunda parte é composta por um sub-pacote que contém uma classe que executa as operações de leitura, escrita e actualização de cada objecto do domínio, usando a tecnologia *JDBC*. Suporta ainda a inserção de uma lista de objectos, e a leitura de um ou de todos os objectos existentes na base de dados.

- **InterfaceWWW** Este pacote contém a classe mais importante, a *SGDwww*. Nesta classe defini-se toda a interacção com as páginas *Web* da base de dados *SGD* usando a biblioteca *HTTPUnit*. Neste pacote podem ser criadas várias classes de interface com a *Web*, em que cada uma sabe interagir com uma ou mais páginas de uma forma específica.

  Neste caso a classe *SGDwww* têm como objectivo a extracção de informação a partir de três páginas da base de dados *SGD*, tendo como informação inicial o nome da ORF.

  O pseudo-código da *SGDwww* é o seguinte:

  1. Para cada ORF contida no texto
  2. Aceder à página inicial a partir do nome da ORF
  3. Extrair a informação e construir os objectos correspondentes
  4. Passar pelas duas páginas seguintes
  5. Extrair a informação e construir os objectos correspondentes
  6. Aceder à base de dados e guardar todos os objectos construídos
  7. Em caso de erro em algum destes passos, guardar o nome da ORF e o erro correspondente. Passar ao ponto 2 com a próxima ORF.

- **Robot** Esta é a classe principal, instanciando o objecto da *interfaceWWW* correspondente à interface *Web* a pesquisar. Nesta classe podem ser chamadas várias interfaces de forma consecutiva. É ainda nesta classe que é feita a configuração dos parâmetros do *proxy-HTTP*.

Utilização

Ao aceder às páginas da base de dados *SGD* com o *Web Spider* foi possível extrair o nome do gene, os nomes alternativos do gene e o *URL* directo da página correspondente nesta base de dados. Estes dados foram utilizados para preencher a tabela *orfgene* da base de dados desenvolvida.

Foi possível ainda para a tabela de relação *translation* fazer a correspondência entre o nome do gene e o nome das proteínas. Para o preenchimento da tabela *protein* foi extraído o

nome da proteína e a sua descrição, sendo necessário mudar de página para extrair a sequência de aminoácidos que a compõe.

Foi ainda possível extrair as anotações do *Gene Ontology* para cada proteína e construir assim as tabelas de ligação entre as proteínas e as três tabelas das ontologias.

4.4.3 Ficheiros auxiliares

Folhas de cálculo

A informação obtida na *Web* a partir de várias bases de dados foi complementada com a informação existente nas folhas de cálculo que o grupo de Ciências Biológicas possuía. Estas continham sequências de *consensus* associadas aos factores de transcrição e informação relativa à tabela *regulation* indicando quais das proteínas eram factores de transcrição e quais os genes regulados. Associada a esta informação existem ainda as referências bibliográficas aos artigos que descreveram cada uma das regulações.

Estas folhas de cálculo foram importadas para a base de dados usando *scripts* na linguagem *Perl*. No apêndice 8.3 pode ser visto um exemplo de um desses *scripts* utilizados, para extrair o *consensus*, a proteína e a referência bibliográfica correspondente.

Estes *scripts* lêem o ficheiro linha a linha, constroem a *query* de inserção com os dados relativos ao objecto existente na base de dados, inserem-no e passam à proxima linha do ficheiro.

Ficheiros de texto

Para obter a informação correspondente às sequências da região promotora dos genes foi utilizada a base de dados da *RSA tools*. Após o pedido do nome de todas as ORF, foi recebido por correio electrónico um ficheiro de texto com todas as sequências promotoras. No apêndice 8.4, é apresentado o *script* utilizado para a leitura deste ficheiro de texto e inserção da informação na base de dados.

4.4.4 Inserção Manual

Embora todos estes métodos de carregamento sejam automáticos e preencham a maioria da base de dados, não garantem a sua manutenção.

Desta forma foi necessário criar um acesso à base de dados por forma a permitir que esta fosse permanentemente actualizada. Para tal, foi criada uma área fornecendo um conjunto de funcionalidades acessíveis através da *Web*. Este acesso é efectuado através de um nome de utilizador e uma palavra de passe.

Esta área permite a inserção, remoção e modificação de praticamente todos os objectos existentes. Desta forma, sempre que existirem novas informações a serem inseridas, estas podem ser inseridas directamente pelos curadores da base de dados. Esta funcionalidade é vantajosa pois permite completar e corrigir a informação existente. Na Figura 4.23, são ilustradas algumas das funcionalidades do *backoffice*, reservadas aos curadores da base de dados.

Na página inicial, aquando da entrada de um utilizador, verifica-se quais os objectos existentes na base de dados com um ou mais campos vazios (Figura 4.24). Estes objectos são depois apresentados numa listagem de forma a permitir ter um panorama geral dos objectos incompletos existentes. É ainda possível aceder directamente à página de actualização desse mesmo objecto simplemente seleccionando no *link* correspondente.

Esta contabilização de objectos incompletos é feita para os objectos mais susceptíveis de ainda não se conhecer toda a informação. Um destes objectos é o das regiões *consensus* de um factor de transcrição, na maior parte das vezes é conhecida a regulação e só posteriormente a região *consensus*. Mesmo sem toda a informação, é criada na base de dados o objecto *consensus* associado à proteína correspondente, tendo uma referência bibliográfica dessa descoberta.

Outra situação comum prende-se com o conhecimento da região de *consensus*, sem o conhecimento da correspondente referência bibliográfica. Uma outra situação é o conhecimento de uma determinada regulação, desconhecendo-se se essa regulação é de activação ou de repressão. Neste caso, existe a correspondência entre o factor de transcrição e o gene, mas o campo indicador do tipo da relação fica vazio.

Uma última situação muito comum tem a ver com a referência bibliográfica relativa às regulações. Por vezes, estas não são conhecidas, tal como acontece com as referências bibliográficas dos *consensus*.

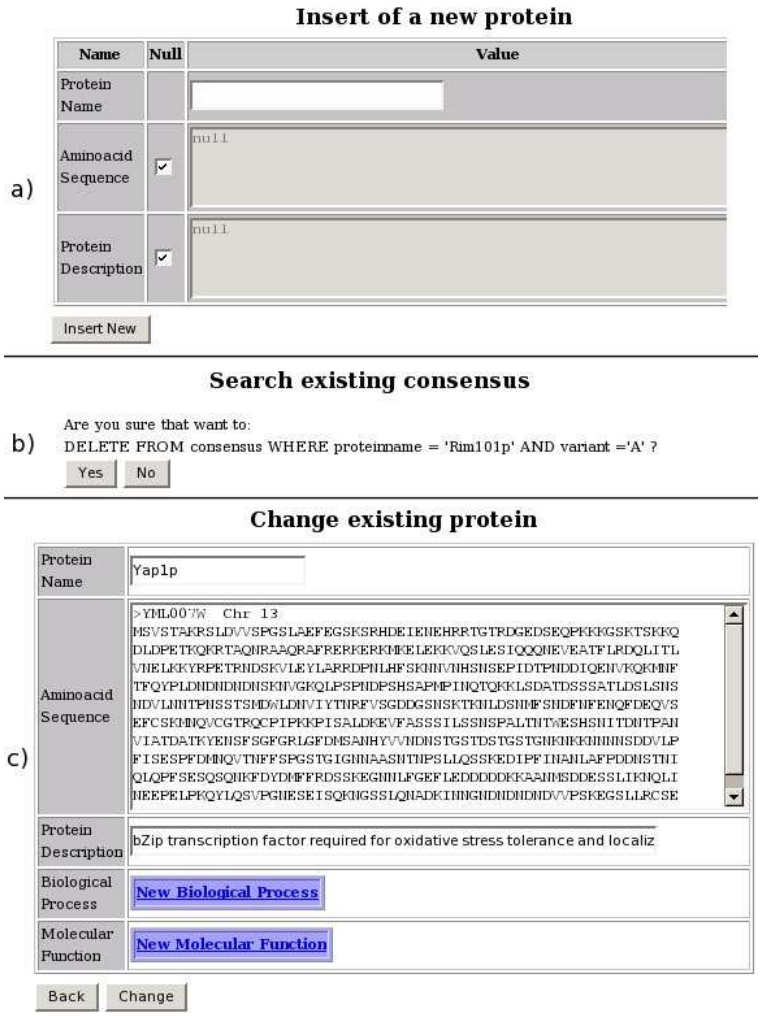


Figura 4.23: a) Interface de inserção de uma proteína. b) Interface de remoção de um *consensus*. c) Interface de modificação da descrição de uma proteína.

4.4.5 Normalização de dados

Após a inserção da informação na base de dados, surgiu o problema da normalização dos dados. Devido a existirem certas convenções para os nomes dos genes, ORF e proteínas, foram construídos alguns *scripts* em *Perl* para a normalização de toda esta informação.

Foi ainda posto um filtro no *backoffice* de forma a normalizar os novos dados inseridos, aquando da inserção ou modificação dos dados. Foi ainda criado um *backoffice* especial para o administrador do sistema (Figura 4.25), com a possibilidade de executar estas normalizações para cada objecto da base de dados de forma automática, verificando inconsistências.

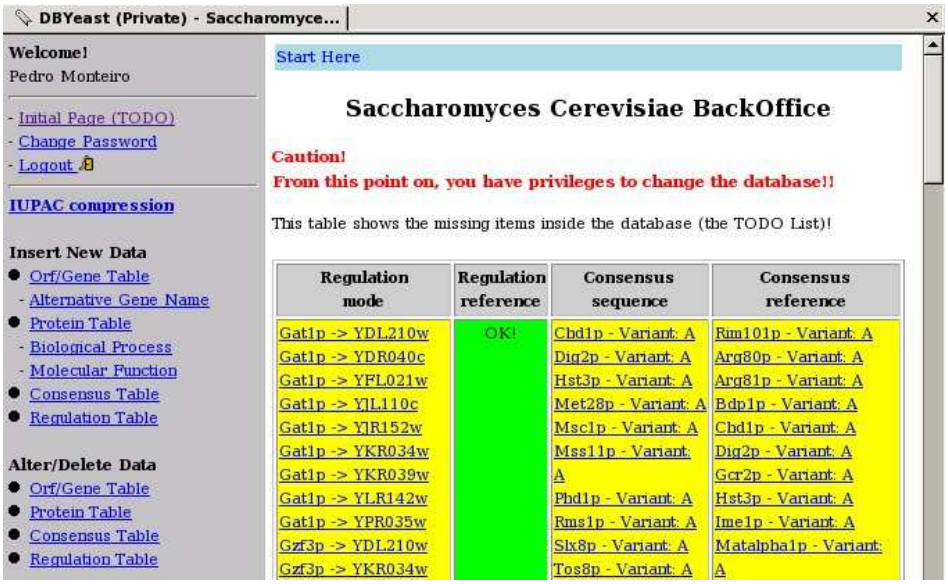


Figura 4.24: Tabela de objectos incompletos na base de dados.

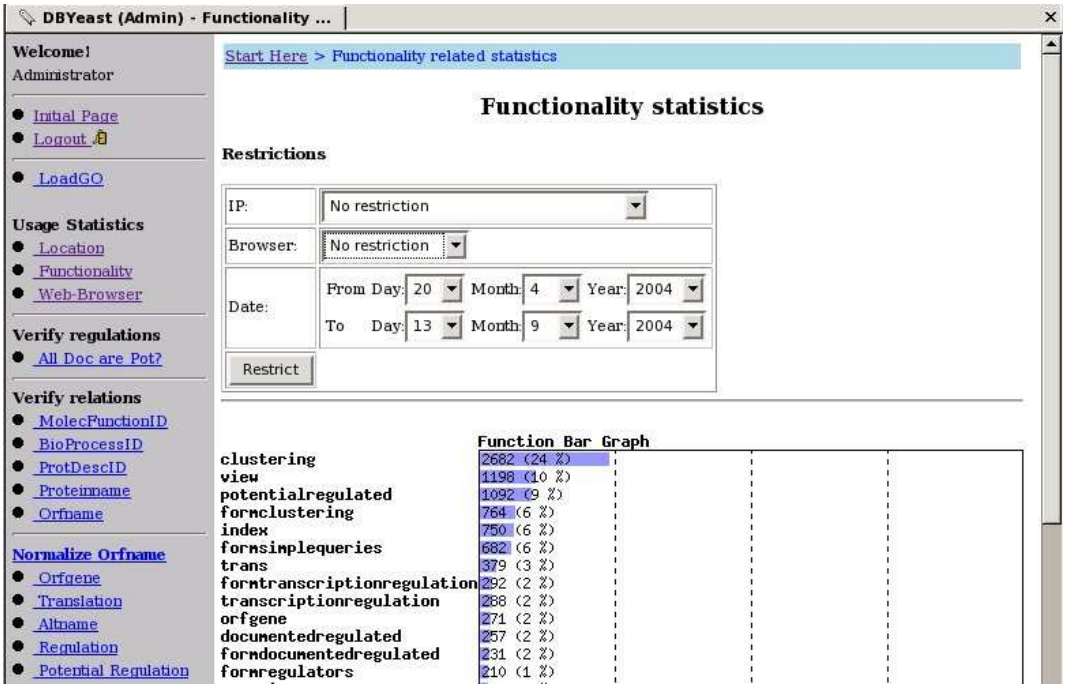


Figura 4.25: Funcionalidades exclusivas do administrador da base de dados.

Como se pode ver na Figura 4.25 é também possível observar estatísticas de acesso ao sistema. Foram criadas três vistas distintas para o fazer, sendo a informação apresentada

através de um gráfico. Existe a vista por funcionalidade, ou seja, qual a funcionalidade mais utilizada, a vista por localização, ou seja, qual o computador remoto que mais acede ao sistema, e a vista por tipo de *browser*, indicando qual o *browser* mais utilizado. Esta última vista é utilizada para modificar a apresentação com o objectivo de satisfazer a maior percentagem de utilizadores.

Em qualquer uma destas vistas, o gráfico de acessos pode ser filtrado usando como restrições a data de acesso e os valores das outras vistas. Desta forma, é possível observar quais as funcionalidades mais acedidas por um determinado grupo de investigação.

Capítulo 5

Funcionalidades implementadas

Actualmente, o sistema de informação contém um conjunto de funcionalidades, que permitem dar resposta a inúmeros problemas normalmente encontrados por quem desenvolve investigação na área da biologia molecular, nesta nova era da genómica.

As funcionalidades implementadas giram em torno do modelo básico apresentado na Figura 5.1. Este modelo representa o facto de os genes codificarem proteínas, e estas proteínas poderem ser factores de transcrição, podendo estes regular a transcrição de um outro conjunto de genes.

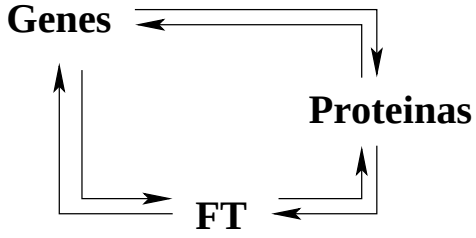


Figura 5.1: Modelo básico da regulação de genes.

Assim, inúmeras questões podem surgir. No sentido directo, a partir de um conjunto de genes pretende-se identificar quais destes codificam factores de transcrição e também quais destes são regulados por estes factores de transcrição. No sentido inverso, a partir de um conjunto de genes pretende-se identificar quais os factores de transcrição que os regulam. Indo um pouco mais longe, podemos também, a partir de um conjunto de genes e um conjunto de factores de transcrição, identificar as regulações entre estes dois conjuntos.

De seguida, são apresentadas as funcionalidades implementadas, descrevendo a sua necessidade e a solução encontrada.

5.1 Simple Queries

Esta funcionalidade permite responder a muitas questões através de acessos directos às tabelas da base de dados, sem a utilização adicional de algoritmos para o tratamento dos dados.

O poder expressivo das respostas a estas questões pode ainda ser aumentado utilizando o símbolo '%' em qualquer campo, significando que essa posição representa qualquer conjunto de caracteres, podendo este conjunto ser vazio. Por exemplo, inserindo no campo **Gene Name** os caracteres '*Gene%*', obter-se-á o *GeneA* e o *GeneZZZ* e, todos os genes cujo nome comece pela palavra '*Gene*'.

Como podemos observar na Figura 5.2, podemos obter informação representada em qualquer dos campos da tabela *orfgene* através do nome da ORF ou do gene.

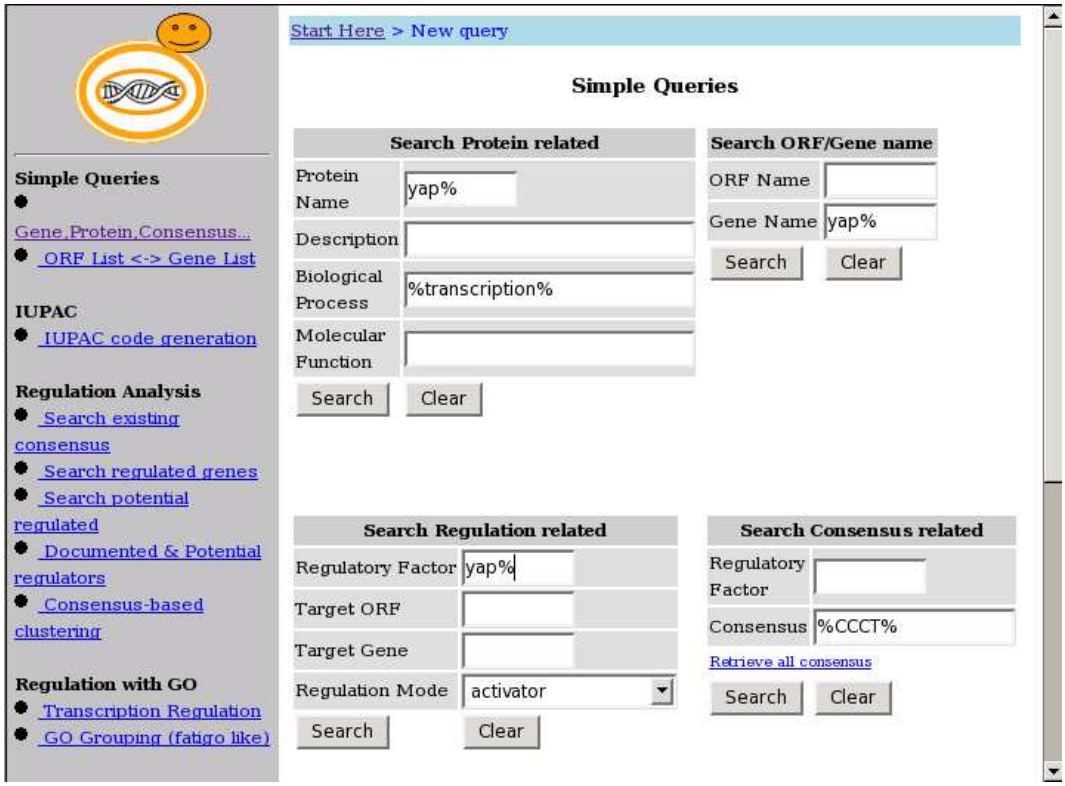


Figura 5.2: Interface para efectuar perguntas simples.

Existem quatro tipos de procuras simples, que reportam informação relacionada com proteínas, com ORF/genes, com factores de transcrição e com *consensus*, respectivamente.

Qualquer que seja o tipo de procura é possível efectuar pesquisas inserindo termos em mais do que um campo, significando que os resultados satisfazem todos os termos inseridos simultâneamente. Assim, numa pesquisa relacionada com proteína, inserindo por exemplo, '*yap*%' no nome da proteína e '%*transcription*%' no processo biológico, obtemos todas as proteínas cujo nome começa por '*yap*' e que estão envolvidas em processos biológicos que contêm a palavra '*transcription*'.

Na Figura 5.3 é apresentado outro tipo de funcionalidade de bastante utilidade. A tradução de nomes de ORF em nomes de genes e vice-versa, através da inserção de uma lista que pode conter simultâneamente nomes de ORF e de genes. A lista inserida é separada em duas, contendo uma os nomes de ORF e a outra o nome dos genes correspondentes.

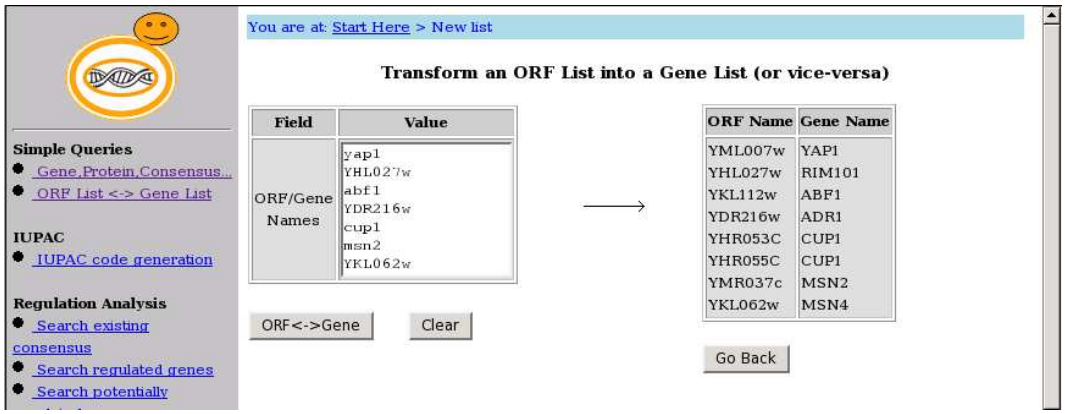


Figura 5.3: Tradução de uma lista de ORF em uma lista de genes e vice-versa.

5.2 Geração de código IUPAC

Esta funcionalidade surge com o objectivo de preencher uma lacuna existente entre as ferramentas disponibilizadas à comunidade científica. Esta nova ferramenta permite obter a melhor compressão para um conjunto de sequências de ADN, através do aumento do alfabeto utilizado para descrever o ADN.

Este novo alfabeto foi apresentado pela *International Union for Pure and Applied Chemistry*, sendo designado por código IUPAC. Cada símbolo neste alfabeto representa um conjunto

de símbolos no alfabeto ADN. A tabela 5.1 apresenta o alfabeto IUPAC assim como a sua correspondência com o alfabeto do ADN.

Alfabeto IUPAC	Alfabeto ADN
W	A ou T
S	C ou G
R	A ou G
Y	T ou C
M	A ou C
K	T ou G
D	A ou T ou G
H	A ou T ou C
V	A ou C ou G
B	T ou G ou C
N	A ou T ou G ou C

Tabela 5.1: Correspondência entre alfabeto IUPAC e o alfabeto ADN.

Esta ferramenta foi desenvolvida no âmbito de um trabalho final de curso realizado por David Nunes e Nuno Mendes [14]. A compressão é efectuada através da adaptação de um algoritmo de minimização lógica denominado ESPRESSO, desenvolvido por Richard Rudell e Alberto Sangiovanni-Vincentelli [15].

5.3 Procura de sequências *consensus*

Esta funcionalidade permite verificar a existência de regiões *consensus* já descritas na base de dados.

Como se pode ver na Figura 5.4, na página de resultados desta funcionalidade, em conjunto com as sequências *consensus*, é apresentado o factor de transcrição que reconhece essa sequência, e quais os genes que estão documentados como sendo regulados por esse factor de transcrição.

Por razões de eficiência, a procura destas regiões *consensus* encontra-se actualmente limitada. A sequência de *consensus* a procurar tem de ter um número de caracteres igual ou

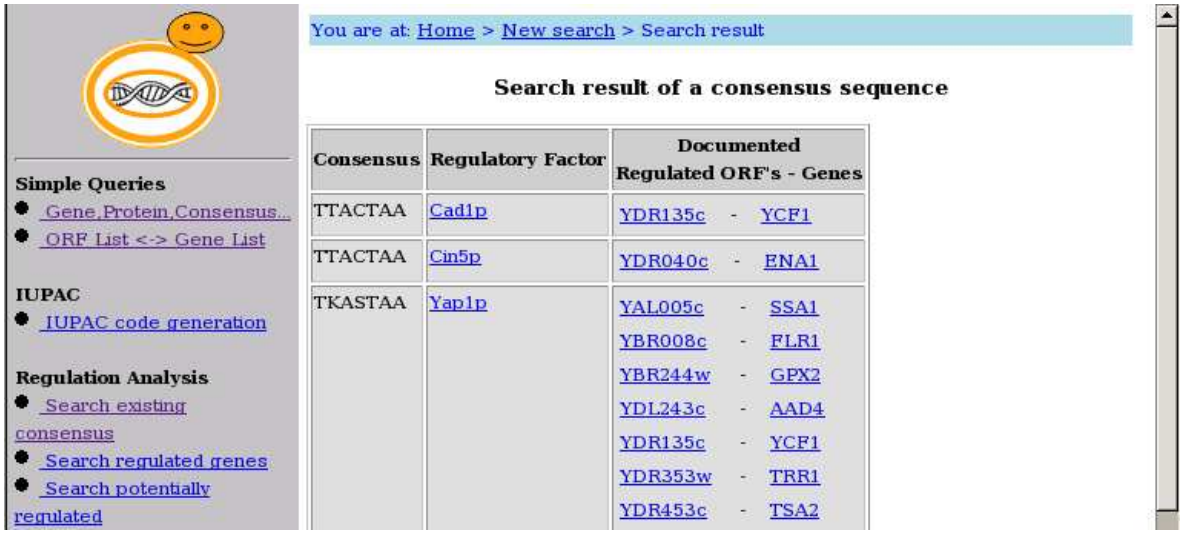


Figura 5.4: Resultados da procura pela sequência *consensus* TTACTAA.

superior ao número de caracteres das sequências *consensus* descritas na base de dados, por forma a ser possível verificar se alguma destas está contida na sequência inserida.

Está prevista a melhoria desta funcionalidade, de forma a permitir a pesquisa com sequências menores do que as sequências *consensus* presentes na base de dados.

Como resultado da pesquisa serão obtidas todas as sequências cujo autómato que as descreve aceita também a sequência inserida. *consensus* que contêm a sequência a pesquisar, ou seja, todas as sequências cujo o autómato gerado contém o autómato gerado pela sequência inserida.

Esta funcionalidade faz uso do alfabeto IUPAC, visto que as sequências de *consensus* que estão inseridas na base de dados, foram descritas utilizando este alfabeto. No exemplo da Figura 5.4, a sequência da pesquisa foi a sequência 'TTACTAA', e na base de dados existiam as sequências TTACTAA e TKACTAA. A sequência que contém o símbolo K, foi desdobrada em duas sequências, TTACTAA e TGACTAA. Assim, o resultado é constituído por todas as sequências *consensus* existentes na base de dados, que geram uma sequência igual à sequência da pesquisa.

5.4 Procura por genes regulados (documentados)

Esta funcionalidade permite, através da inserção de uma lista de factores de transcrição, obter a lista de genes que estão documentados como sendo regulados, por algum dos factores de transcrição, como ilustrado na Figura 5.5.

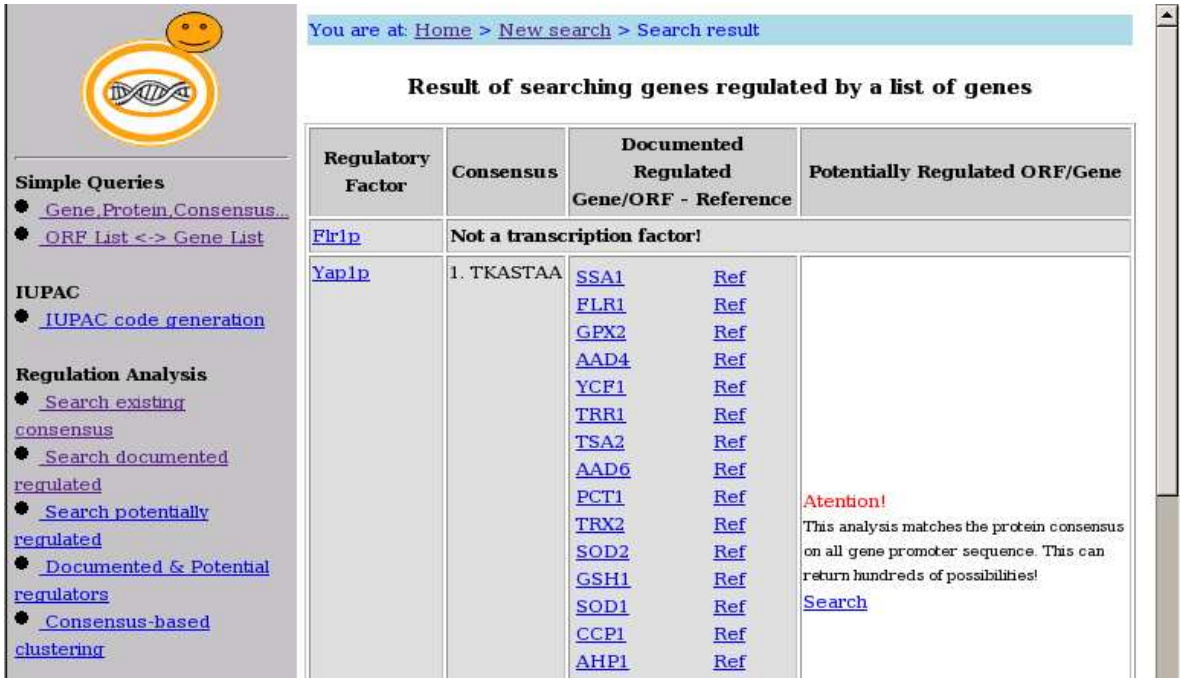


Figura 5.5: Lista de genes documentados como sendo regulados pela lista de factores de transcrição inserida.

No entanto, o formulário de entrada não é restrito à inserção de nomes de proteínas. É também possível inserir os nomes dos genes que codificam as proteínas em análise.

No caso de algumas das proteínas inseridas não serem um factor de transcrição, é apresentada uma mensagem indicativa de que não o são.

O pseudo-código desta funcionalidade é o seguinte:

1. Para cada nome contido na lista;
2. Verificar é um nome de um gene; e se o for, substituir pelo nome da proteína codificada;
3. Para cada proteína da nova lista, verificar se é um factor de transcrição;
4. Em caso verdadeiro, extrair o gene regulado e a referência bibliográfica correspondente;

É ainda inserida uma coluna extra, permitindo pesquisar quais os genes potencialmente regulados por cada factor de transcrição. Esta funcionalidade é descrita de seguida.

5.5 Procura por genes regulados (potenciais)

Esta funcionalidade tem como objectivo identificar quais os genes potencialmente regulados por um factor de transcrição. Ao contrário da funcionalidade anterior, esta permite a inserção de apenas um factor de transcrição, visto que a lista de resultados para cada factor de transcrição pode ser bastante vasta, como podemos verificar na Figura 5.6.

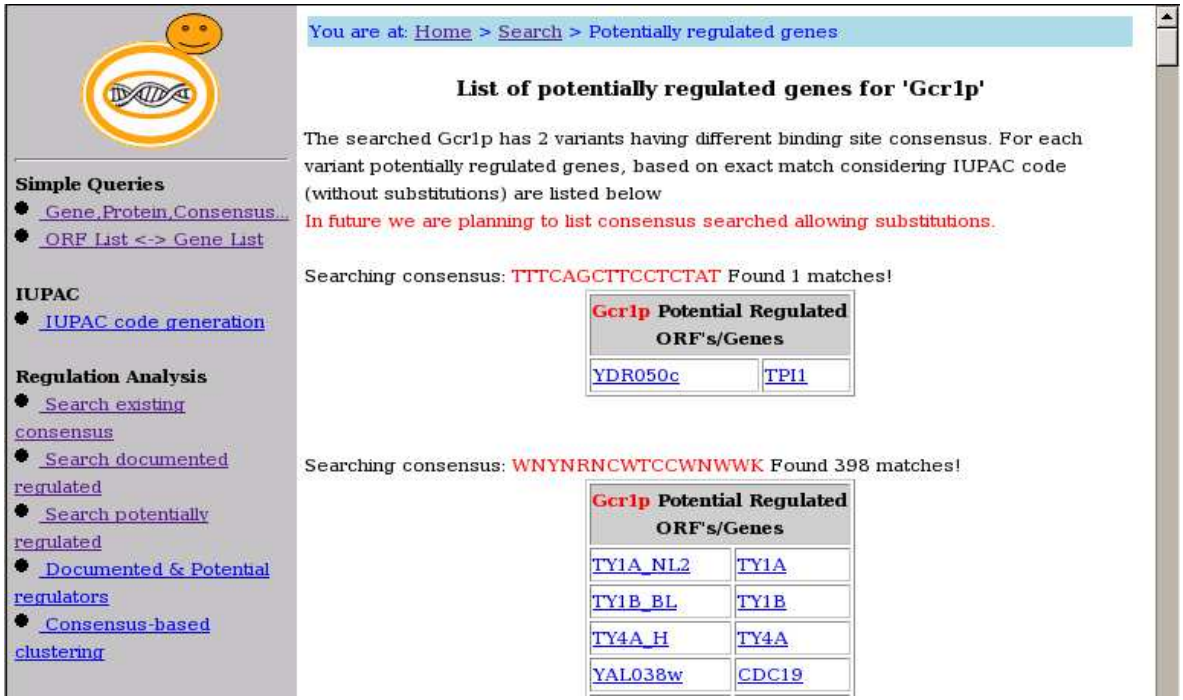


Figura 5.6: Lista de genes potencialmente regulados por um determinado factor de transcrição.

O pseudo-código que descreve esta funcionalidade é o seguinte:

1. Verificar se o nome inserido é de um gene; se for, substituir pelo nome da proteína codificada;

2. Verificar se a proteína tem regiões de *consensus* associadas, documentadas na base de dados;

3. Se tiver, para cada região de *consensus*:

   (a) Verificar se essa região está presente na região promotora de algum dos genes presentes na base de dados;

   (b) Devolver a lista de genes que têm a região de *consensus* presente.

Esta funcionalidade está, actualmente desenvolvida sem a utilização de métodos de pesquisa com erros. Pode acontecer por vezes, que um determinado factor de transcrição tenha uma região de *consensus* associada, mas que o sistema não a encontre presente na região promotora de nenhum dos genes da base de dados, por essa região conter um caracter errado.

A pesquisa, utilizando por exemplo uma função como a distância de edição para medir a semelhança entre duas cadeias de caracteres não foi desenvolvida, visto que existem regiões de *consensus* que já são muito genéricas, sendo encontradas em quase todas as regiões promotoras de genes, tornando a interpretação da lista de resultados quase impossível.

Actualmente, o grupo de Ciências Biológicas do IST está a efectuar uma verificação das regiões de *consensus* existentes na base de dados, na tentativa de corrigir algumas das descrições dos *consensus* no alfabeto IUPAC.

Apenas quando existir uma lista de *consensus* sem erros, será possível ter uma lista credível de genes potencialmente regulados. Nesta altura é plausível adaptar a pesquisa actual a uma pesquisa que contemple a existência de erros, permitindo obter uma lista de genes regulados por terem uma região promotora com afinidade suficiente para a ligação do factor de transcrição.

5.6 Procura por FTs documentados/potenciais

Esta funcionalidade permite seguir o esquema da Figura 5.1 no sentido inverso das setas, ou seja, a partir dos genes regulados, identificar quais os factores de transcrição que os regulam.

O resultado desta pesquisa é apresentado de duas formas distintas: em forma de tabela, indicando tanto os factores de transcrição reguladores documentados, como os potenciais, como ilustrado na Figura 5.7; e em forma de imagem, apresentando uma representação dos promotores dos vários genes inseridos, e a ligação dos vários factores de transcrição nos promotores, como ilustrado na Figura 5.8.

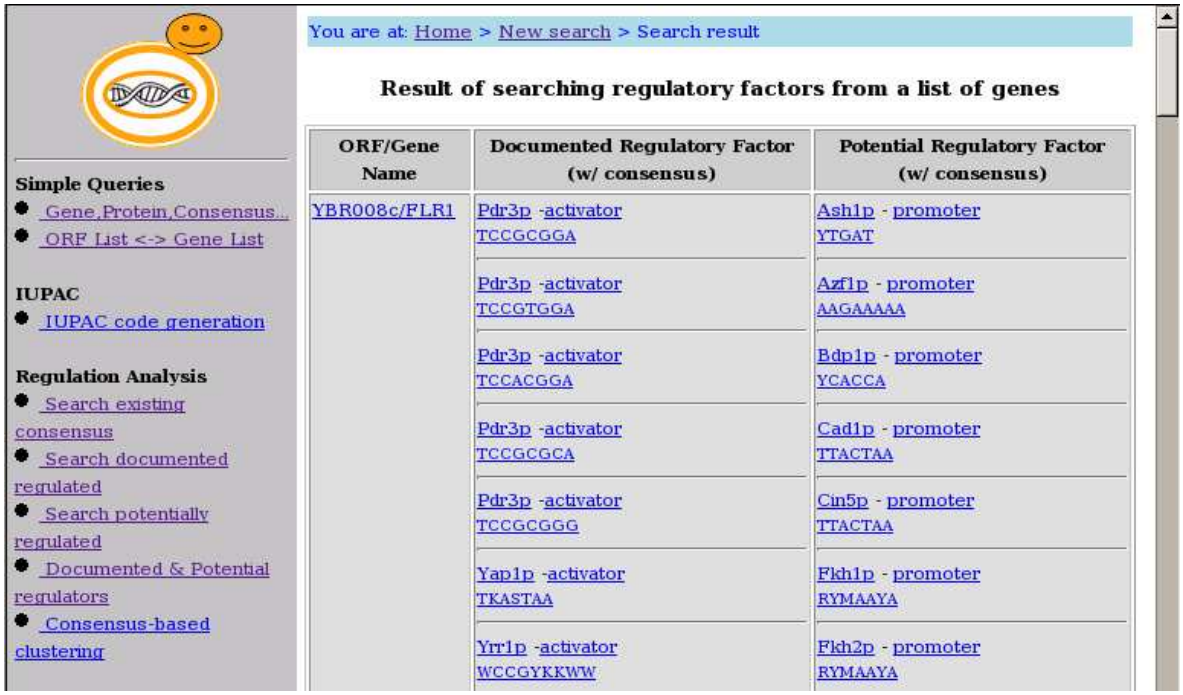


Figura 5.7: Factores de transcrição que estão documentados como reguladores e que potencialmente regulam o gene FLR1.

O resultado através da tabela, é composto por três colunas. A primeira, indicando os nomes dos genes regulados. A segunda, indicando os factores de transcrição que estão documentados como regulando o gene correspondente. Para cada um destes factores de transcrição é indicado o modo de regulação (activador, repressor ou ambos). Por fim, a terceira coluna indica os factores de transcrição potenciais, em que para cada um deles, é possível ver a localização da região de *consensus* na região promotora do gene.

Na representação através de uma imagem, é possível eliminar alguns dos factores de transcrição. Esta funcionalidade é bastante vantajosa nos casos em que existem factores de transcrição com regiões de *consensus* genéricas e que sobrecarregam a imagem escondendo outras regulações.

O facto de serem apenas apresentados, na imagem, os factores de transcrição potenciais, tem a ver com o facto de não dispormos da posição com que um factor de transcrição, com uma determinada região de *consensus*, se liga à região promotora de um gene.

Por outro lado, visto que temos as regiões promotoras dos genes e as regiões de *consensus*

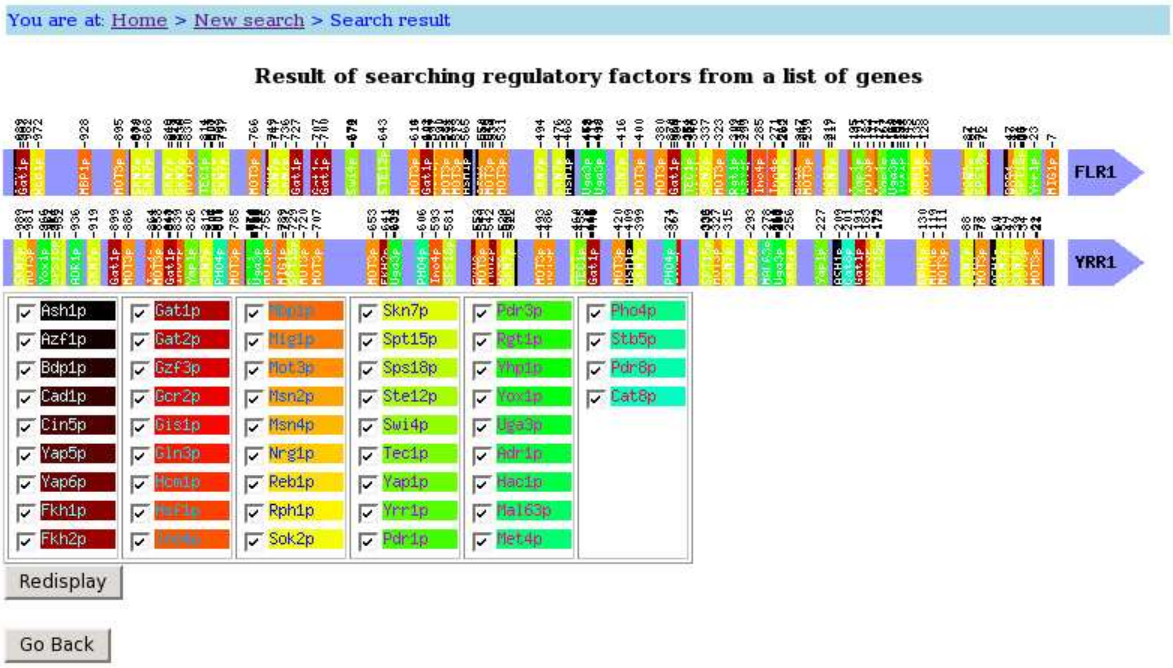


Figura 5.8: Representação da ligação potencial dos factores de transcrição, existentes na base de dados, ao promotor dos genes FLR1 e YRR1.

dos factores de transcrição, podíamos para cada regulação documentada, o local onde as regiões de *consensus* apareciam na região promotora dos genes. No entanto, esta apresentação não seria rigorosa, visto que uma sequência de *consensus* pode aparecer várias vezes na região promotora e não existe informação sobre qual das posições é a correcta, ou seja, qual das posições é a considerada pelo autor do artigo que descreve a regulação.

Pode acontecer ainda que nenhuma das regiões de *consensus* de um determinado factor de transcrição, esteja presente na região promotora dos genes. Esta situação deve-se à região de *consensus* não se encontrar bem descrita ou devido à procura efectuada não permitir erros.

5.7 Consensus based clustering

A partir de experiências de Microarrays[1] ou outras experiências laboratoriais, obtem-se uma lista de genes que são expressos quando as células são submetidas a determinadas condições,

[1]Vector de moléculas de ADN que permite realizar em paralelo inúmeras experiências de hibridação. Com este dispositivo é possível monitorizar o nível de expressão de milhares de genes em simultâneo.

como por exemplo, um determinado factor de stress químico.

Os genes compilados nestas experiências podem ter as mais variadas funções na célula. Uma parte destes genes vai codificar proteínas que são factores de transcrição, ou seja, que por sua vez irão regular outros genes.

O objectivo desta funcionalidade é identificar a partir de uma lista de genes ou das proteínas codificadas, quais os que codificam factores de transcrição e quais destes estão envolvidos na regulação (documentada e potencial) de genes contidos na lista inicial. No caso das regulações documentadas é apresentada a referência bibliográfica correspondente.

Na Figura 5.9 é apresentado um exemplo de identificação de factores de transcrição e dos genes regulados. Neste exemplo, podemos verificar os casos de auto-regulação dos genes PDR3 e YRR1.

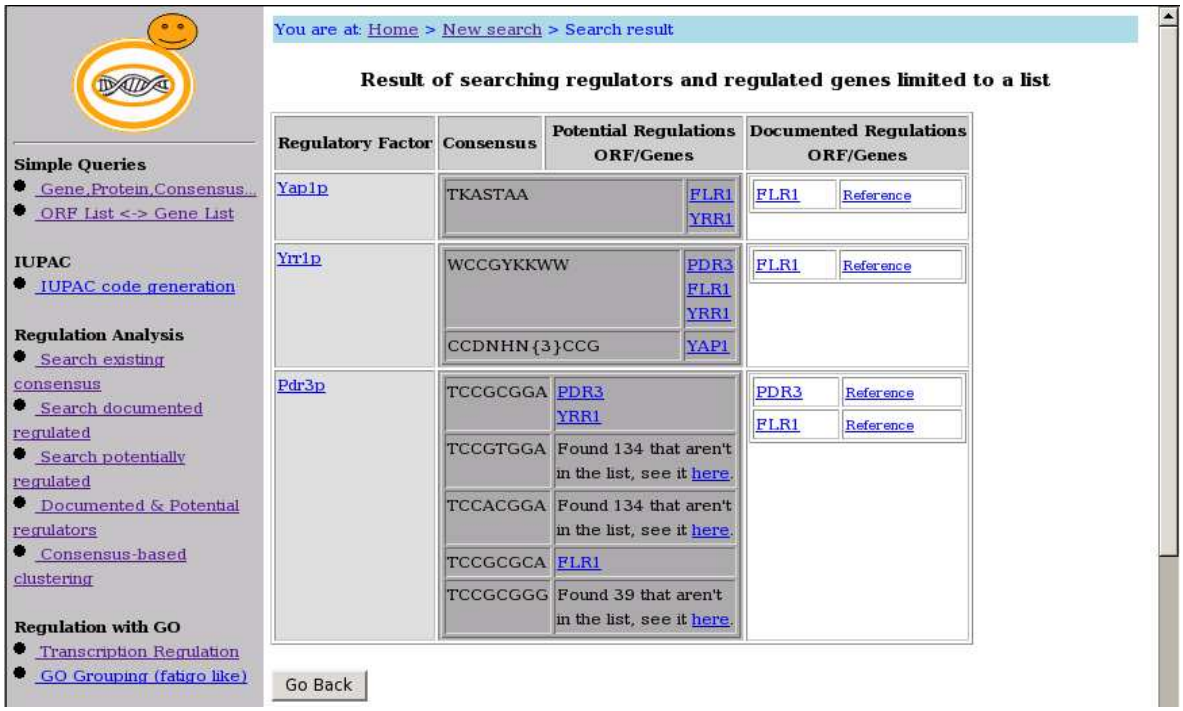


Figura 5.9: Regulações existentes entre os genes da lista YAP1, FLR1, YRR1 e PDR3.

O pseudo-código que descreve esta funcionalidade é o seguinte:

1. Para cada gene da lista inicial;

2. Verificar se codifica um factor de transcrição;

3. Em caso afirmativo, apresentar as sequências de *consensus* associadas e os genes potencialmente regulados através de cada sequência de *consensus*;

4. Apresentar também os genes que estão documentados como sendo regulados por esse factor de transcrição, assim como a referência bibliográfica correspondente.

5.8 Transcription Regulation

Esta funcionalidade surgiu inicialmente como uma evolução da anterior, numa tentativa de uniformização das últimas quatro pesquisas apresentadas, possibilitando inúmeras pesquisas.

Os resultados obtidos podem ser agrupados por grupos de termos do *Gene Ontology Consortium*, podendo estes grupos pertencer à ontologia de processo biológico ou de função molecular. Cada uma das listas, reguladores e regulados, pode ser agrupada de uma forma independente, usando ontologias distintas.

Como podemos ver na Figura 5.10, o formulário de entrada apresenta várias opções que serão descritas de seguida.

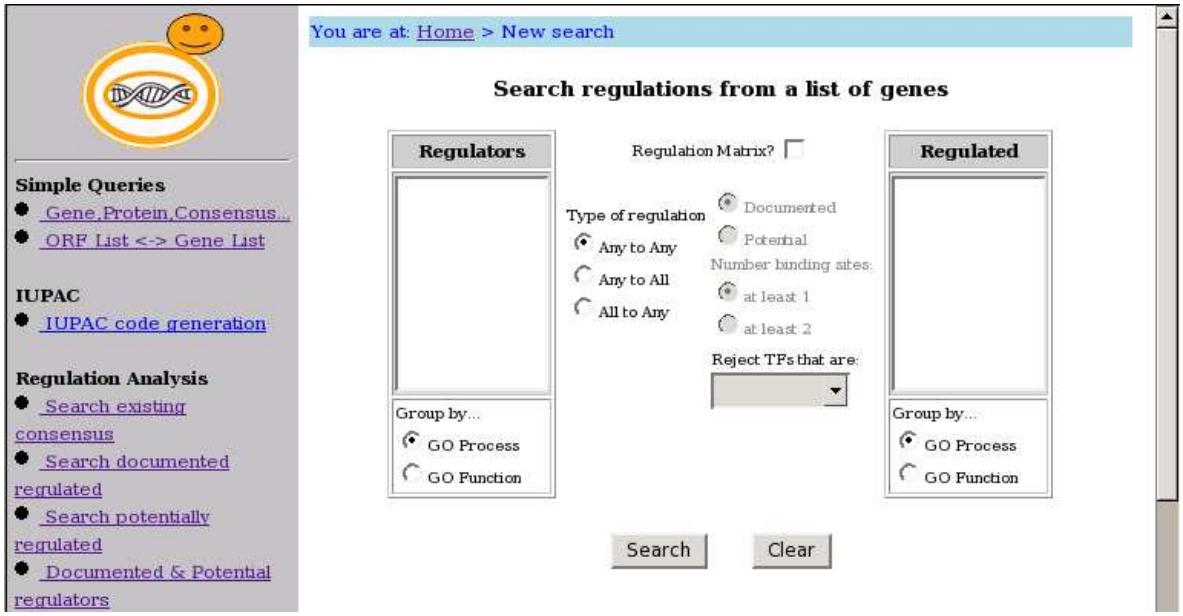


Figura 5.10: Formulário da funcionalidade *Transcription Regulation*.

À primeira vista é possível verificar que existe uma separação entre a lista de reguladores, factores de transcrição, e de regulados, genes. Nesta separação está implicita uma funciona-

lidade. Se for inserida uma lista de factores de transcrição e uma lista de genes regulados, a pesquisa de regulações é efectuada apenas entre as duas listas. Se a lista dos factores de transcrição for deixada vazia, significa que a pesquisa é entre a lista presente de genes e todos os factores de transcrição existentes na base de dados.

Por outro lado, se a lista de genes regulados estiver vazia, significa que a pesquisa de regulações é efectuada entre a lista de factores de transcrição inserida e todos os genes existentes na base de dados.

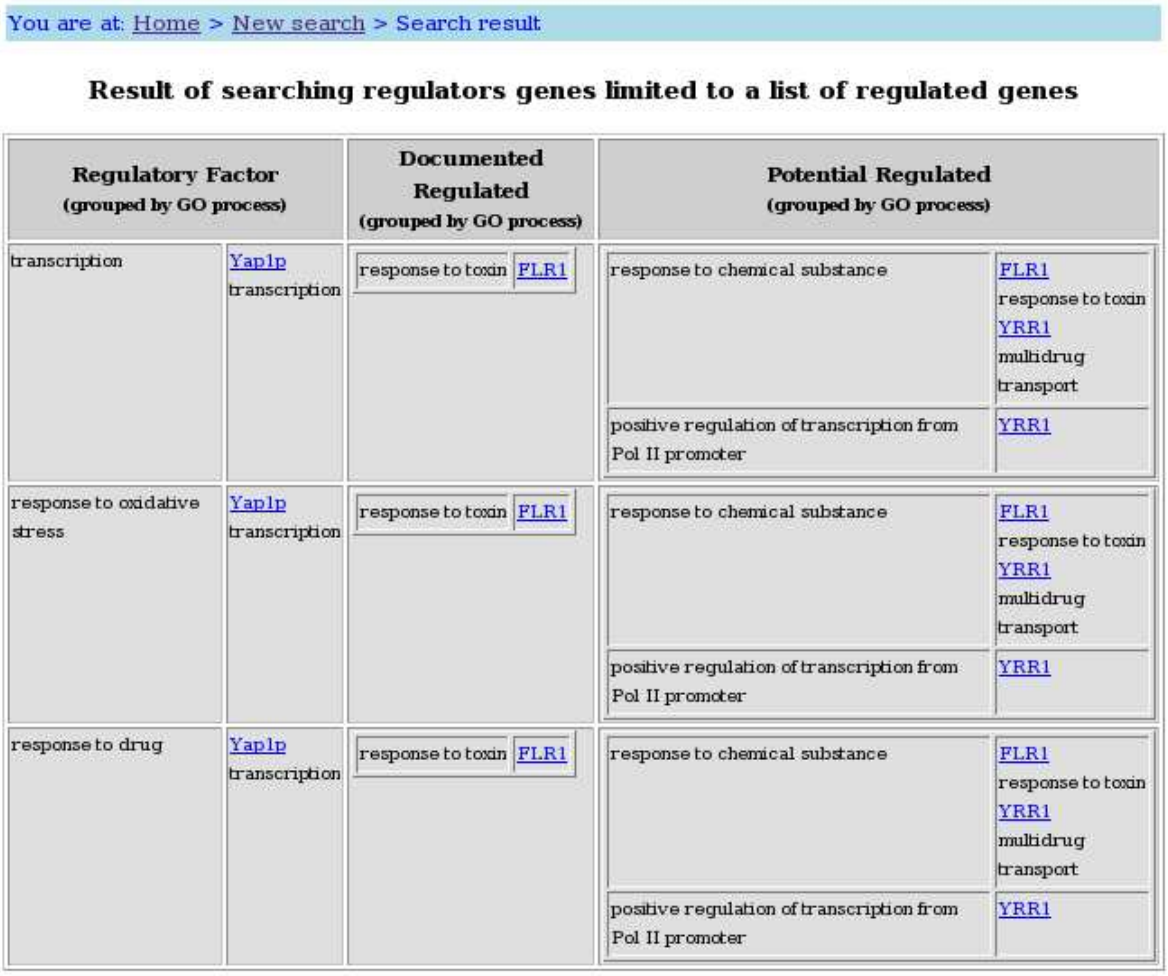
Foi ainda adicionada uma particularidade nesta pesquisa. É possível pesquisar por regulações em que todos os factores de transcrição da lista, têm de regular todos os subconjunto de genes; ou pesquisar por regulações em que qualquer subconjunto da lista de factores de transcrição, tem de regular todos os genes inseridos na lista de genes regulados; ou ainda fazer uma pesquisa em que qualquer subconjunto da lista de factores de transcrição regula qualquer subconjunto da lista de genes.

Como podemos verificar na Figura 5.11, o resultado desta pesquisa é apresentado usando três colunas.

Na primeira coluna é apresentado o agrupamento dos factores de transcrição pela ontologia correspondente. Na segunda coluna, são apresentados os genes documentados como sendo regulados pelo grupo de factores de transcrição correspondente. Na terceira coluna, são apresentados os genes potencialmente regulados pelo grupo de factores de transcrição correspondente. No topo de cada coluna, é indicada qual a ontologia utilizada para o agrupamento dos genes ou factores de transcrição.

Cada agrupamento de factores de transcrição, corresponde a uma linha principal da tabela. As células correspondentes à intersecção de cada linha com cada uma das três colunas, estão divididas em dois. No lado esquerdo é apresentado o termo da ontologia que mais especificamente representa o conjunto de factores de transcrição ou genes apresentado do lado direito. Por baixo de cada proteína ou gene, está ainda o termo mais específico associado.

Esta funcionalidade é a que mais pode evoluir dentro do contexto desta base de dados, visto que está dirigida para a inferência de redes de regulação. No entanto, o recurso a tabelas não é a melhor solução para a apresentação dos resultados, pois o cruzamento de tanta informação torna-se ilegível para uma pesquisa com mais de cinco ou dez factores de transcrição e genes.



You are at: Home > New search > Search result

Result of searching regulators genes limited to a list of regulated genes

Regulatory Factor (grouped by GO process)		Documented Regulated (grouped by GO process)		Potential Regulated (grouped by GO process)	
transcription	Yap1p transcription	response to toxin	FLR1	response to chemical substance	FLR1 response to toxin YRR1 multidrug transport
				positive regulation of transcription from Pol II promoter	YRR1
response to oxidative stress	Yap1p transcription	response to toxin	FLR1	response to chemical substance	FLR1 response to toxin YRR1 multidrug transport
				positive regulation of transcription from Pol II promoter	YRR1
response to drug	Yap1p transcription	response to toxin	FLR1	response to chemical substance	FLR1 response to toxin YRR1 multidrug transport
				positive regulation of transcription from Pol II promoter	YRR1

Figura 5.11: Pesquisa de regulações utilizando as ontologias do *Gene Ontology Consortium*.

A apresentação em forma de tabela é também uma condicionante do algoritmo utilizado para a apresentação dos resultados. Esta funcionalidade torna-se um pouco lenta quando é pesquisado um número razoável de genes ou factores de transcrição.

A solução passa por encontrar uma forma mais simples e legível para a apresentação dos resultados. Provavelmente, no futuro, será utilizada uma forma gráfica.

O pseudo-código que descreve esta funcionalidade é o seguinte:

1. Eliminar da lista dos factores de transcrição todas as proteínas repetidas e também as que não são factores de transcrição, e se esta lista for vazia, considerar todos os factores de transcrição descritos na base de dados;

2. Eliminar da lista de genes regulados os repetidos, e se esta lista for vazia, considerar todos os genes descritos na base de dados;

3. Agrupar os factores de transcrição pelos termos da ontologia escolhida;

4. Para cada factor de transcrição pesquisar as regulações documentadas e as regulações potenciais;

5. Para cada conjunto de genes regulados por um factor de transcrição, agrupar pelos termos da ontologia escolhida;

5.8.1 Matriz de regulações

É ainda possível escolher outra funcionalidade no formulário apresentado na Figura 5.11, a apresentação das regulações representadas sob a forma de uma matriz.

O resultado é um ficheiro com os valores separados por vírgulas [2] que pode ser aberto numa folha de cálculo, em que as colunas representam todos os genes inseridos e as linhas os factores de transcrição. Cada célula de intersecção entre um gene e um factor de transcrição pode ter o valor um ou zero, consoante exista uma regulação, ou não entre os dois.

Como é possível ver na Figura 5.10, para aceder a esta funcionalidade, é necessário seleccionar a opção *Regulation Matrix*. É possível escolher uma pesquisa pelas regulações documentadas ou pelas regulações potenciais. No caso de ser escolhido as regulações documentadas, a pesquisa pode ser filtrada eliminando os factores de transcrição que actuam como activadores ou como repressores. No caso de serem escolhidas as regulações potenciais, a pesquisa pode ser filtrada, escolhendo apenas as regulações que têm pelo menos dois locais de ligação na região promotora dos genes, ou então escolhendo as regulações que têm pelo menos um local de ligação.

Esta funcionalidade não usa o agrupamento pelos termos das ontologias do *Gene Ontology Consortium*. No entanto, é uma forma rápida e simples de representar as regulações entre uma lista de factores de transcrição e uma lista de genes.

Estas matrizes são posteriormente utilizadas por algoritmos de *biclustering* para, de uma forma automática, identificar possíveis redes de regulação.

[2] Em Inglês, *Comma Separated Values* (CSV)

Capítulo 6

Avaliação do Sistema

6.1 Regulações documentadas vs. potenciais

O sistema desenvolvido disponibiliza duas visões para uma regulação que envolva um factor de transcrição e um gene: a regulação potencial e a regulação documentada, como foi referido na secção 2.5.1. Estas duas visões podem apresentar diferenças significativas.

A regulação potencial, é obtida através do emparelhamento de cadeias de caracteres correspondentes às sequências de *consensus* reconhecidas pelo factor de transcrição, e a região promotora de um gene. Estas regulações confirmam que o factor de transcrição se liga fisicamente à região promotora do gene. Noutro tipo de regulações, um factor de transcrição pode fazer parte de um complexo proteico, iniciando ou inibindo a regulação de um gene, sem estar directamente em contacto com a região promotora do gene.

A regulação documentada mostra que um factor de transcrição regula de alguma forma a transcrição de um gene, não especificando se a regulação é efectuada pela ligação do factor de transcrição à região promotora do gene ou ao complexo proteíco constituído por outros factores de transcrição.

Tendo estas duas visões em mente, efectuámos uma contagem do número de regulações documentadas, do número de regulações potenciais e do número de regulações documentadas que também estavam descritas como potenciais. O resultado pode ser observado na Figura 6.1.

Observa-se então que apenas 32.34% (aproximadamente um terço) das regulações documentadas estão simultaneamente descritas como documentadas e potenciais. No entanto, estão representadas no sistema quase dezoito vezes mais regulações potenciais do que re-

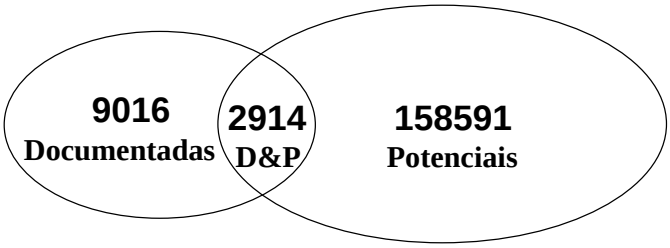


Figura 6.1: Relação entre as regulações documentadas e potenciais.

gulações documentadas.

Esta baixa percentagem de regulações documentadas que são simultaneamente classificados como regulações potenciais levanta algumas questões sobre a sua justificação. Uma das justificações passa pelo facto, referido anteriormente, de nem todas as regulações documentadas serem regulações em que o factor de transcrição se liga directamente à região promotora do gene. Por outro lado, apesar de no cálculo das regulações potenciais se efectuar o desdobramento do código IUPAC das sequências de *consensus*, permitindo a procura de regulações com alguma ambiguidade, ainda não é possível efectuar procuras usando distâncias de edição [16].

Relativamente ao elevado número de regulações potenciais descritas actualmente no sistema, é também possível apontar algumas justificações. Uma das justificações passa pelo tamanho (número de pares de bases) considerado para a região promotora. A região promotora dos genes não tem um tamanho fixo. No entanto, é usual considerar-se para a região promotora até 1000 pares de bases a montante, ou seja, permitir a sobreposição da região promotora com as ORFs a montante, aumentando assim a região considerada para ligação dos factores de transcrição. Esta opção deve-se ao facto de nesta região poderem ligar-se não só os factores de transcrição que dão início à regulação como também factores de transcrição que têm o papel de atenuadores e/ou de acentuadores da transcrição.

6.2 Funcionalidades inovadoras

Muitas são as bases de dados já existentes com inúmeros conceitos representados. Estes conceitos podem ir desde genes, sequências promotoras, factores de transcrição, regiões de *consensus*, e outros. No entanto, poucas são as bases de dados que relacionam todos estes conceitos, deixando assim algum espaço para o desenvolvimento de novas funcionalidades.

Este novo sistema surge assim como um complemento às bases de dados já existentes, apresentando novas funcionalidades com um foco especial nas relações entre os conceitos, não deixando de apresentar os conceitos existentes noutros sistemas.

6.2.1 Procura por genes regulados (documentados)

Esta funcionalidade permite ao utilizador inserir uma lista de factores de transcrição e obter de forma expedita uma tabela com a lista dos genes documentados e regulados por cada um desses factores de transcrição, bem como as referências bibliográficas que justificam cada uma das regulações.

6.2.2 Procura por genes regulados (potenciais)

Esta funcionalidade permite ao utilizador inserir um factor de transcrição e obter a lista de genes potencialmente regulados, tendo em conta a sequência de *consensus* reconhecida pelo factor de transcrição em causa. O utilizador tem ainda a possibilidade de visualizar o local de ligação desse factor de transcrição na região promotora de cada gene.

6.2.3 Procura por FTs documentados/potenciais

Esta funcionalidade permite ao utilizador inserir uma lista de genes e obter duas listas de factores de transcrição: uma com os factores de transcrição que estão documentados como regulando cada um dos genes inseridos e outra com os factores de transcrição que contêm uma região de *consensus* que se liga à região promotora de cada um dos referidos genes.

Esta funcionalidade permite ainda, apenas para as regulações potenciais, a construção em tempo real de uma imagem, que permite a visualização da distribuição espacial da ligação de cada factor de transcrição na região promotora dos genes especificados.

6.2.4 Consensus Based Clustering

Esta funcionalidade permite uma conjugação das funcionalidades descritas anteriormente, podendo o utilizador inserir uma lista de genes e de factores de transcrição. Fica a cargo do sistema a separação entre o que são factores de transcrição e o que são genes regulados, apresentando posteriormente quais os factores de transcrição presentes na lista inserida e

quais os genes presentes na lista regulados (documentados e potenciais) por esses factores de transcrição.

6.2.5 Geração de código IUPAC

Esta funcionalidade apesar de ter sido desenvolvida no âmbito de um trabalho final de curso [14], não deixa de representar uma funcionalidade ainda inexistente nas actuais ferramentas. Utiliza o alfabeto IUPAC para descrever de uma forma compacta um conjunto de sequências descritas utilizando o alfabeto ADN.

Por exemplo, é possível representar as sequências `TCCGTGGG`, `TCCGTGGA`, `TCCATGGA`, `TCCGTGGG`, `TCCGCGGG` apenas utilizando as sequências `TCCRTGGA` e `TCCGYGGG` descritas utilizando o alfabeto IUPAC.

6.3 Utilização do sistema

Em termos da utilização do sistema, este tem vindo a ser testado, de uma forma intensiva, pelo grupo de ciências biológicas do IST, enquanto parceiro de desenvolvimento. No entanto, o sistema foi também recentemente disponibilizado a alguns grupos de investigação internacionais com actividade científica no domínio da levedura.

As funcionalidades actualmente existentes no sistema foram desenvolvidas de acordo com as necessidades e sugestões do grupo de ciências biológicas do IST. Este tipo de desenvolvimento, tem os seus aspectos positivos, na medida em que é um desenvolvimento bastante focado com uma aplicação bastante concreta, e tem também aspectos negativos, visto que a especialização de uma funcionalidade pode não satisfazer as necessidades de outros grupos de utilização.

A divulgação a nível internacional deste sistema vai permitir obter contribuições e sugestões críticas que suportem o crescimento estável do sistema desenvolvido. Foram já recebidos comentários favoráveis dos grupos internacionais de investigação que estão a testar o sistema, tais como da Washington University in St. Louis, Université Catholique de Louvain, Université Libre de Bruxelles, tendo ainda acessos da École Normale Supérieure em França e da Universidade de Stanford. Estes comentários vêm reforçar a importância que um sistema como este pode ter para a comunidade científica.

A primeira versão deste sistema já foi submetida para publicação na conferência *Yeast Genetics and Molecular Biology, 2005*, uma das maiores conferências internacionais para o organismo biológico levedura. Esta publicação pretende apresentar o sistema dando ênfase ao seu carácter inovador relativamente aos conteúdos biológicos. Dentro em breve será submetida uma publicação à revista *Nucleic Acids Research* onde o sistema será apresentado não só focando o conteúdo biológico mas também focando o sistema de informação desenvolvido e as suas capacidades de modelação dos conceitos envolvidos.

Capítulo 7

Conclusões e Trabalho Futuro

Um dos problemas existentes na área de bioinformática, em especial no domínio dos sistemas de informação, é o facto de existirem muitas bases de dados, cada uma especializada em fornecer informação relativa a um determinado conceito, mas não existir nenhuma que relacione todos estes conceitos, com acesso livre a todos os interessados. Devido à dificuldade em manter um sistema deste tipo, estes têm tido um dos três seguintes destinos: primeiro, podem deixar de ser actualizados por terem sido fruto de um trabalho académico pontual; segundo, podem conseguir manter-se actualizados mas acabam por se especializar apenas em alguns conceitos, levando o investigador, no decurso de um determinado processo, a usar vários sistemas para cada uma das etapas; terceiro, alguns destes sistemas acabam por ter interesse para o sector privado e acabam por apresentar restrições de acesso.

O sistema desenvolvido tem como objectivo o suporte e integração de vários conceitos da biologia molecular e as suas relações, permitindo não só a integração dos conceitos existentes em outros sistemas, mas também a integração de informação do grupo de ciências biológicas do IST, permitindo centralizar e fornecer uma visão integrada de toda a informação existente. O sistema permitiu assim a substituição de inúmeras folhas de cálculo na partilha de informação, constituindo um importante instrumento de trabalho não só para o grupo de ciências biológicas do IST, como para a toda a comunidade que investiga o organismo biológico levedura.

Durante todo o desenvolvimento, o sistema teve como principais avaliadores os membros do grupo de ciências biológicas do IST, que testou o sistema constantemente, fornecendo críticas construtivas e a necessária visão biológica na interpretação dos resultados obtidos. Este

sistema foi recentemente difundido junto de outros grupos de investigação, tendo recebido comentários muito positivos, o que deixa prever uma boa evolução futura.

O sistema revelou-se também uma valiosa plataforma de suporte à integração e teste de novos algoritmos de bioinformática, como é por exemplo o gerador de código IUPAC.

O desenvolvimento futuro deste sistema terá três direcções principais. A primeira é a redução do número de regulações potenciais, visto que muitas destas não têm significado biológico, bem como a integração de novos algoritmos para a procura de motivos, tais como *bi-clustering* ou outros métodos de análise de padrões, permitindo assim expandir o leque de funcionalidades disponibilizadas.

Em segundo lugar, é necessário a melhoria do sistema de actualização e integração de novos dados, visto que nova informação está constantemente a surgir. Muitos destes dados necessitam de ser pré-processados de forma a facilitar a utilização em tempo real de algumas das funcionalidades disponibilizadas.

Por último, um dos objectivos mais arrojados deste sistema é permitir a inferência e visualização de redes de regulação de genes. Para tal será necessário adicionar uma componente algorítmica complexa, bem como resolver toda a problemática da visualização destas redes de regulação de forma a permitir ao utilizador uma navegação fácil. A obtenção destas redes será um dos maiores desafios deste sistema.

Capítulo 8

Apêndice

8.1 IDBAccess

```
<?php
function stringDate() {
    $now = getdate();
    $date  = $now['year'];
    $date .= "-".$now['mon'];
    $date .= "-".$now['mday'];
    $date .= " ".$now['hours'];
    $date .= ":".$now['minutes'];
    $date .= ":".$now['seconds'];
    return $date;
}


/* Generic Interface for a DB Access
 * calls the specific class that represents the
 * desired connection
 */
class IDBAccess {
    var $db;
    var $_logquery;

    function IDBAccess($type = "mysql")
    {
        $dbclass = "DB$type";
        $this->db = new $dbclass;
        $this->_logquery = false;
    }
```

```
function openRead()
{
    include 'config.php';
    $host = $dbconfig['dbHost'];
    $dbname = $dbconfig['dbName'];
    $user = $dbconfig['dbPlainUser'];
    $pass = $dbconfig['dbPlainPass'];
    $this->db->open($host, $dbname, $user, $pass);
    $this->_logquery = $dbconfig['logquery'];
}

function getWriteUserLogin()
{
    include 'config.php';
    return $dbconfig['dbAdminUser'];
}

function getWriteUserPass()
{
    include 'config.php';
    return $dbconfig['dbAdminPass'];
}

function openWrite()
{
    include 'config.php';
    $host = $dbconfig['dbHost'];
    $dbname = $dbconfig['dbName'];
    $user = $dbconfig['dbAdminUser'];
    $pass = $dbconfig['dbAdminPass'];
    $this->db->open($host, $dbname, $user, $pass);
    $this->_logquery = $dbconfig['logquery'];
}

function close()
{
    $this->db->close();
    unset($this->db);
}

function insertUpdate($query)
{
    $this->logThisQuery($query);
```

```
        return $this->db->insertUpdate($query);
    }

    function query($name,$query)
    {
        $this->logThisQuery($query);
        return $this->db->query($name,$query);
    }

    function num_rows($name)
    {
        return $this->db->num_rows($name);
    }

    function nextObject($name)
    {
        return $this->db->nextObject($name);
    }

    function getObject($name,$query)
    {
        $this->logThisQuery($query);
        return $this->db->getObject($name,$query);
    }

    function freeResult($name)
    {
        $this->db->freeResult($name);
    }

    function dataSeek($name,$rowNumber)
    {
        return $this->db->dataSeek($name,$rowNumber);
    }

    function logThisQuery($query) {
        if (!$this->_logquery)
            return;
        if (strpos($query,"log"))
            return;
        if (@$_SESSION['user'] && @$_SESSION['user']->username)
             $user = $_SESSION['user']->username;
        else $user = "anonymous";
        $date = stringDate();
```

```
        $q = addslashes($query);
        $this->db->insertUpdate("INSERT INTO log_db VALUES('','$user','$date','$q')");
    }
}


/* Specific SQL connection - MySQL Functions
 * Handle multiple results simultaneous
 *  - using $hash{'result_name'}
 */
class DBmysql {
    var $_link;
    var $_result;

    function open($host, $dbname, $user, $pass)
    {
        /* Open a mysql connection */
        $this->_link = mysql_connect($host,$user,$pass)
        or die("Could not connect: " . mysql_error());

        /* Selects the mysql database */
        mysql_select_db($dbname)
        or die("Could not select database $dbname");
    }

    function close()
    {
        mysql_close($this->_link);
    }

    function getObject($name,$query)
    {
        $field = FALSE;
        $result = mysql_query($query);
        if (@mysql_num_rows($result) > 0) {
            $obj = mysql_fetch_object($result);
            $field = $obj->$name;
            mysql_free_result($result);
        }
        return $field;
    }

    function insertUpdate($query)
    {
```

```
        return mysql_query($query);
    }

    function error()
    {
        return mysql_error($this->_link);
    }

    function query($name,$query)
    {
        if (isset($this->_result{$name}))
            unset($this->_result{$name});
        $this->_result{$name} = mysql_query($query);
        if ($this->_result{$name} &&
            (mysql_num_rows($this->_result{$name}) > 0)) {
            return TRUE;
        } else {
            unset($this->_result{$name});
            return FALSE;
        }
    }

    function num_rows($name)
    {
        if (isset($this->_result{$name})) {
            return mysql_num_rows($this->_result{$name});
        }
        else return 0;
    }

    /* returns an array with the next obj fields
     * return FALSE otherwise
     * $name - result name (handling multiple results)
     */
    function nextObject($name)
    {
        if (!isset($this->_result{$name}))
            return FALSE;
        $obj = mysql_fetch_array($this->_result{$name},MYSQL_ASSOC);
        return $obj;
    }

    function freeResult($name)
    {
```

```
        if (isset($this->_result{$name}))
            mysql_free_result($this->_result{$name});
        unset($this->_result{$name});
    }

    function dataSeek($name,$rowNumber)
    {
        return mysql_data_seek($this->_result{$name},$rowNumber);
    }
}


/* Specific SQL connection - postgresSQL Functions
 * Handle multiple results simultaneous
 *  - using $hash{'result_name'}
 */
class DBpostgres {
    var $_link;
    var $_result;

    function open($host, $dbname, $user, $pass)
    {
        $connectionString  = "host=$host ";
        $connectionString .= "user=$user ";
        $connectionString .= "password=$pass ";
        $connectionString .= "dbname=$dbname";

        /* Open a postgres connection */
        $this->_link = pg_connect($connectionString)
        or die("Could not connect: ".pg_last_error());
    }

    function close()
    {
        pg_close($this->_link);
    }

    function getObject($name,$query)
    {
        $field = FALSE;
        $result = pg_query($this->_link, $query);
        if (@pg_num_rows($result) > 0) {
            $obj = pg_fetch_object($result, 0);
            $field = $obj->$name;
```

```
            pg_free_result($result);
        }
        return $field;
    }

    function insertUpdate($query)
    {
        return pg_query($this->_link, $query);
    }

    function error()
    {
        return pg_last_error($this->_link);
    }

    function query($name,$query)
    {
        if (isset($this->_result{$name}))
            unset($this->_result{$name});
        $this->_result{$name} = pg_query($this->_link, $query);
        if ($this->_result{$name} &&
            (pg_num_rows($this->_result{$name}) > 0)) {
            return TRUE;
        } else {
            unset($this->_result{$name});
            return FALSE;
        }
    }

    function num_rows($name)
    {
        if (isset($this->_result{$name})) {
            return pg_num_rows($this->_result{$name});
        }
        else return 0;
    }

    /* returns an array with the next obj fields
     * return FALSE otherwise
     * $name - result name (handling multiple results)
     */
    function nextObject($name)
    {
        if (!isset($this->_result{$name}))
```

```
            return FALSE;
        $obj = _fetch_array($this->_result{$name},MYSQL_ASSOC);
        return $obj;
    }

    function freeResult($name)
    {
        if (isset($this->_result{$name}))
            pg_free_result($this->_result{$name});
        unset($this->_result{$name});
    }

    function dataSeek($name,$rowNumber)
    {
        return pg_result_seek($this->_result{$name}, $rowNumber);
    }
}
?>
```

8.2 Exemplo de utilização da classe IDBAccess

```
foreach ($initgenes as $gene) {
    $o = normalizeGene($gene);
    $q  = "SELECT orfname, genename FROM orfgene ";
    $q .= "WHERE orfname = '$o'";
    if ($db->query("orf", $q)) {
        $row = $db->nextObject("orf");
        $o = normalizeGene($row['orfname']);
        $g = normalizeGene($row['genename']);
        if (strcasecmp($g,"Unknown"))
            $genes[$o] = $g;
        else $genes[$o] = $o;
    } else {
        $q  = "SELECT orfname,genename FROM orfgene ";
        $q .= "WHERE genename = '$o'";
        if ($db->query("gene",$q)) {
            $row = $db->nextObject("gene");
            $o = normalizeGene($row['orfname']);
            $g = normalizeGene($row['genename']);
            if (strcasecmp($g,"Unknown"))
                $genes[$o] = $g;
            else $genes[$o] = $o;
        } else {
            $q  = "SELECT orfname,alternativename ";
```

```
            $q .= "FROM altname WHERE alternativename ='$o'";
            if ($db->query("altgene",$q)) {
                $row = $db->nextObject("altgene");
                $o = normalizeGene($row['orfname']);
                $g = normalizeGene($row['alternativename']);
                if (strcasecmp($g,"Unknown"))
                    $genes[$o] = $g;
                else $genes[$o] = $o;
            }
        }
    }
}
```

8.3 Ficheiro extracção de *consensus*

```
#!/usr/bin/perl -w

use strict;
use DBI;

my @lines;
while(<>){
        chomp;
        push @lines, $_;
}

my $dbh = DBI->connect(
            'dbi:mysql:biology',
            'ptgm',
            'ptgm',
            {
              RaiseError => 1,
              AutoCommit => 0
            }) || die "Database connection not made: $DBI::errstr";


foreach my $line (@lines) {
  $line =~ /^[^;]*;([^;]*);([^;]*);([^;]*);([^;]*);([^;]*)$/;

  my ($variant, $consense, $protname, $function, $reference) = ($1,$2,$3,$4,$5);

#  print "[$variant] [$consense] [$protname] [$function] [$reference]\n";

  my $consdataid = 0;
```

```
if (length($reference) > 1) {
  eval {
    my $sql = qq{ INSERT INTO reference VALUES ( ? , ? ) };
    my $sth = $dbh->prepare( $sql );
    $sth->execute('',$reference);
    $dbh->commit();
  };
  my $sql = qq{ SELECT referenceID FROM reference where reference = ? };
  my $sth = $dbh->prepare($sql);
  $sth->execute($reference);
  my $referenceid;
  $sth->bind_columns(undef, \$referenceid);
  $sth->fetch();
  $sth->finish();

  eval {
    my $sql = qq{ INSERT INTO consensedata VALUES ( ? , ? ) };
    my $sth = $dbh->prepare( $sql );
    $sth->execute('',$referenceid);
    $dbh->commit();
  };
  $sql = qq{ SELECT consdataID FROM consensedata where referenceID = ? };
  $sth = $dbh->prepare($sql);
  $sth->execute($referenceid);
  $sth->bind_columns(undef, \$consdataid);
  $sth->fetch();
  $sth->finish();
}

my $sql = qq{ INSERT INTO consense VALUES ( ? , ? , ? , ? ) } ;
eval {
  my $sth = $dbh->prepare( $sql );
  $sth->execute($consense,$variant,$protname,$consdataid);
  $dbh->commit();
};
if( $@ ) {
  warn "Database error: $@\n";
  $dbh->rollback(); #just die if rollback is failing
}

eval {
  my $sql = qq{ UPDATE activation SET activationmode = ? where proteinname = ?} ;
  my $sth = $dbh->prepare( $sql );
  $sth->execute($function,$protname);
```

```
    $dbh->commit();
  };
  if( $@ ) {
    warn "Database error: $@\n";
    $dbh->rollback(); #just die if rollback is failing
  }

}
$dbh->disconnect();
```

8.4 Ficheiro extracção de promotores

```
#!/usr/bin/perl -w

use strict;
use DBI;

my $dbh = DBI->connect(
            'dbi:mysql:biology',
            'biologyadmin',
            '1atat1mp',
            {
              RaiseError => 1,
              AutoCommit => 0
            }) || die "Database connection not made: $DBI::errstr";

my @sequences = split(">", `cat result.2003_11_26.181829.txt`);
shift @sequences;
foreach my $s (@sequences) {
  my $seq = ">".$s;
  $seq =~ /^>(\S+)\s+/;
  my $orfgene = $1;

  my ($orf,$promseq) = ("","");
  eval {
    my $sql = qq{ SELECT orfname,promotersequence FROM orfgene where orfname = ? OR genename = ? };
    my $sth = $dbh->prepare($sql);
    $sth->execute($orfgene, $orfgene);
    $sth->bind_columns(undef,\$orf,\$promseq);
    $sth->fetch();
    $sth->finish();
  };
  eval {
    my $sql = qq{ UPDATE orfgene SET promotersequence = ? where orfname = ? } ;
```

```
    my $sth = $dbh->prepare( $sql );
    $sth->execute($seq,$orf);
    $dbh->commit();
  };
  if( $@ ) {
    warn "Database error: $@\n";
    $dbh->rollback(); #just die if rollback is failing
  }
  print "ORF: $orf\n$seq\n\n";
}
$dbh->disconnect();
```

8.5 Código SQL

Código SQL 8.1 Código SQL para a criação da tabela *orfgene*

```
CREATE TABLE `orfgene` (
  `orfname` varchar(10) NOT NULL,
  `genename` varchar(10) default NULL,
  `url` varchar(100) default NULL,
  `genesequence` text,
  `promotersequence` text,
  `retrotransposon` enum('N','Y') NOT NULL default 'N',
  PRIMARY KEY  (`orfname`),
  KEY `genename` (`genename`)
) TYPE=MyISAM;
```

Código SQL 8.2 Código SQL para a criação da tabela *altname*

```
CREATE TABLE `altname` (
  `orfname` varchar(10) NOT NULL,
  `alternativename` varchar(10) NOT NULL,
  PRIMARY KEY  (`orfname`,`alternativename`),
  KEY `alternativename` (`alternativename`)
) TYPE=MyISAM;
```

Código SQL 8.3 Código SQL para a criação da tabela *translation*

```
CREATE TABLE `translation` (
  `orfname` varchar(10) NOT NULL,
  `proteinname` varchar(10) NOT NULL,
  PRIMARY KEY  (`orfname`,`proteinname`),
  KEY `proteinname` (`proteinname`)
) TYPE=MyISAM;
```

Código SQL 8.4 Código SQL para a criação da tabela *protein*

```
CREATE TABLE `protein` (
  `proteinname` varchar(10) NOT NULL,
  `aminoacidsequence` text,
  `protdescID` int(11) default NULL,
  PRIMARY KEY  (`proteinname`)
) TYPE=MyISAM;
```

Código SQL 8.5 Código SQL para a criação da tabela *protdesc*

```
CREATE TABLE `protdesc` (
  `protdescID` int(11) NOT NULL auto_increment,
  `description` varchar(240) NOT NULL,
  PRIMARY KEY  (`protdescID`),
  UNIQUE KEY `description` (`description`)
) TYPE=MyISAM;
```

Código SQL 8.6 Código SQL para a criação da tabela *regulation*

```
CREATE TABLE ‘regulation‘ (
  ‘proteinname‘ varchar(10) NOT NULL,
  ‘orfname‘ varchar(10) NOT NULL,
  ‘regulationmode‘ varchar(40) default NULL,
  ‘regulationdataID‘ int(11) NOT NULL default ’0’,
  PRIMARY KEY  (‘proteinname‘,‘orfname‘),
  KEY ‘orfname‘ (‘orfname‘)
) TYPE=MyISAM;
```

Código SQL 8.7 Código SQL para a criação da tabela *regulationdata*

```
CREATE TABLE ‘regulationdata‘ (
  ‘regulationdataID‘ int(11) NOT NULL auto_increment,
  ‘referenceID‘ int(11) default NULL,
  ‘evidencecodeID‘ varchar(5) default NULL,
  PRIMARY KEY  (‘regulationdataID‘),
  UNIQUE KEY ‘U1‘ (‘referenceID‘,‘evidencecodeID‘)
) TYPE=MyISAM;
```

Código SQL 8.8 Código SQL para a criação da tabela *reference*

```
CREATE TABLE ‘reference‘ (
  ‘referenceID‘ int(11) NOT NULL auto_increment,
  ‘reference‘ varchar(150) NOT NULL,
  PRIMARY KEY  (‘referenceID‘),
  UNIQUE KEY ‘reference‘ (‘reference‘)
) TYPE=MyISAM;
```

Código SQL 8.9 Código SQL para a criação da tabela *evidencecode*

```
CREATE TABLE ‘evidencecode‘ (
  ‘code‘ varchar(5) NOT NULL,
  ‘definition‘ varchar(100) NOT NULL,
  ‘examples‘ text NOT NULL,
  PRIMARY KEY  (‘code‘)
) TYPE=MyISAM;
```

Código SQL 8.10 Código SQL para a criação da tabela *functionlist*

```
CREATE TABLE ‘functionlist‘ (
  ‘ID‘ varchar(15) NOT NULL,
  ‘proteinname‘ varchar(10) NOT NULL,
  PRIMARY KEY  (‘proteinname‘,‘ID‘),
  KEY ‘ID‘ (‘ID‘)
) TYPE=MyISAM;
```

Código SQL 8.11 Código SQL para a criação da tabela *processlist*

```
CREATE TABLE ‘processlist‘ (
  ‘ID‘ varchar(15) NOT NULL,
  ‘proteinname‘ varchar(10) NOT NULL,
  PRIMARY KEY  (‘proteinname‘,‘ID‘),
  KEY ‘ID‘ (‘ID‘)
) TYPE=MyISAM;
```

Código SQL 8.12 Código SQL para a criação da tabela *componentlist*

```
CREATE TABLE ‘componentlist‘ (
  ‘ID‘ varchar(15) NOT NULL,
  ‘proteinname‘ varchar(10) NOT NULL,
  PRIMARY KEY  (‘proteinname‘,‘ID‘),
  KEY ‘ID‘ (‘ID‘)
) TYPE=MyISAM;
```

Código SQL 8.13 Código SQL para a criação da tabela *consensus*

```
CREATE TABLE ‘consensus‘ (
  ‘consensus‘ varchar(50) NOT NULL,
  ‘variant‘ varchar(5) NOT NULL default ’-’,
  ‘proteinname‘ varchar(10) NOT NULL,
  ‘consdataID‘ int(11) default ’0’,
  PRIMARY KEY  (‘proteinname‘,‘variant‘),
  KEY ‘consensus‘ (‘consensus‘)
) TYPE=MyISAM;
```

Código SQL 8.14 Código SQL para a criação da tabela *consensusdata*

```
CREATE TABLE ‘consensusdata‘ (
  ‘consdataID‘ int(11) NOT NULL auto_increment,
  ‘referenceID‘ int(11) default NULL,
  PRIMARY KEY  (‘consdataID‘),
  UNIQUE KEY ‘referenceID‘ (‘referenceID‘)
) TYPE=MyISAM;
```

Código SQL 8.15 Código SQL para a criação da tabela *potentialregulation*

```
CREATE TABLE ‘potentialregulation‘ (
  ‘consensus‘ varchar(50) NOT NULL,
  ‘orfname‘ varchar(10) NOT NULL,
  ‘ID‘ int(11) NOT NULL auto_increment,
  PRIMARY KEY  (‘consensus‘,‘orfname‘),
  UNIQUE KEY ‘ID‘ (‘ID‘)
) TYPE=MyISAM;
```

Código SQL 8.16 Código SQL para a criação da tabela *potentialregulationpos*

```
CREATE TABLE ‘potentialregulationpos‘ (
  ‘ID‘ int(11) unsigned NOT NULL default ’0’,
  ‘pos‘ int(11) unsigned NOT NULL default ’0’,
  ‘len‘ int(11) unsigned NOT NULL default ’0’,
  PRIMARY KEY  (‘ID‘,‘pos‘)
) TYPE=MyISAM;
```

Código SQL 8.17 Código SQL para a criação da tabela *potentialregulationposreverse*

```
CREATE TABLE ‘potentialregulationposreverse‘ (
  ‘ID‘ int(11) unsigned NOT NULL default ’0’,
  ‘pos‘ int(11) unsigned NOT NULL default ’0’,
  ‘len‘ int(11) unsigned NOT NULL default ’0’,
  PRIMARY KEY  (‘ID‘,‘pos‘)
) TYPE=MyISAM;
```

Código SQL 8.18 Código SQL para a criação da tabela *function*

```
CREATE TABLE `function` (
  `ID` varchar(15) NOT NULL,
  `function` varchar(200) NOT NULL,
  `depth` tinyint(4) default NULL,
  PRIMARY KEY  (`ID`),
  UNIQUE KEY `function` (`function`)
) TYPE=MyISAM;
```

Código SQL 8.19 Código SQL para a criação da tabela *process*

```
CREATE TABLE `process` (
  `ID` varchar(15) NOT NULL,
  `process` varchar(165) NOT NULL,
  `depth` tinyint(4) default NULL,
  PRIMARY KEY  (`ID`),
  UNIQUE KEY `process` (`process`)
) TYPE=MyISAM;
```

Código SQL 8.20 Código SQL para a criação da tabela *component*

```
CREATE TABLE `component` (
  `ID` varchar(15) NOT NULL,
  `component` varchar(200) NOT NULL,
  `depth` tinyint(4) default NULL,
  PRIMARY KEY  (`ID`),
  UNIQUE KEY `component` (`component`)
) TYPE=MyISAM;
```

Código SQL 8.21 Código SQL para a criação da tabela *functionparents*

```
CREATE TABLE ‘functionparents‘ (
  ‘ID‘ varchar(15) NOT NULL default ’’,
  ‘sonID‘ varchar(15) NOT NULL default ’’,
  PRIMARY KEY  (‘ID‘,‘sonID‘)
) TYPE=MyISAM;
```

Código SQL 8.22 Código SQL para a criação da tabela *processparents*

```
CREATE TABLE ‘processparents‘ (
  ‘ID‘ varchar(15) NOT NULL default ’’,
  ‘sonID‘ varchar(15) NOT NULL default ’’,
  PRIMARY KEY  (‘ID‘,‘sonID‘)
) TYPE=MyISAM;
```

Código SQL 8.23 Código SQL para a criação da tabela *componentparents*

```
CREATE TABLE ‘componentparents‘ (
  ‘ID‘ varchar(15) NOT NULL default ’’,
  ‘sonID‘ varchar(15) NOT NULL default ’’,
  PRIMARY KEY  (‘ID‘,‘sonID‘)
) TYPE=MyISAM;
```

Bibliografia

[1] C. S. H. Laboratory, The promoter database of saccharomyces cerevisiae. *http://cgsigma.cshl.org/jian/*, 1998.

[2] P. E. Hodges, et al, The yeast proteome database (ypd): a model for the organization and presentation of genome-wide functional data. http://www.proteome.com/ypdhome.html. *Nucleic Acids Research*, Volume 27, pp. 69–73, 1999.

[3] S. University, Saccharomyces cerevisiae genome database. *http://www.yeastgenome.org/*, 2004.

[4] Biobase, Transfac. *http://www.gene-regulation.com/pub/databases.html#transfac*, 2004.

[5] G. O. Consortium, The gene ontology (go) database and informatics resource. *Nucleic Acids Research*, Volume 32, pp. D251–D261, 2004.

[6] J. Watson e F. Crick, A structure for deoxyribose nucleic acid. *Nature*, Volume 171, pp. 737–738, Abril 1953.

[7] G. M. Cooper e R. E. Hausman, *The Cell: A Molecular Approach, Third Edition*, Sinauer Associates, Inc., June 2003.

[8] T. F. Consortium, The flybase database of the drosophila genome projects and community literature. http://flybase.org. *Nucleic Acids Research*, Volume 31, pp. 172–175, 2003.

[9] J. A. Blake, et al, Mgd: The mouse genome database. *Nucleic Acids Research*, Volume 31, pp. 193–195, 2003.

[10] G. O. Consortium, Gene ontology consortium. *http://www.geneontology.org/*, 1998.

[11] J. V. Helden, B. André, e J. Collado-Vides, A web site for the computational analysis of yeast regulatory sequences. *Yeast*, Volume 16, No. 2, pp. 177–187, 2000.

[12] R. Gold, Httpunit. *http://httpunit.sourceforge.net*, 2004.

[13] Sun, Jdbc technology. *http://java.sun.com/j2se/1.4.2/docs/guide/jdbc/index.html*, 2004.

[14] N. Mendes e D. Nunes, Geração de código iupac, Relatório Técnico, INESC-ID, 2004.

[15] R. Rudel e A. Sangiovanni-Vicentelli, Multiple-valued minimization for pla optimization. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, Volume CAD-6, No. 5, pp. 727–751, September 1987.

[16] V. Levenstein, Binary codes capable of correcting insertions and reversals. *Sov. Phys. Dokl*, Volume 10, pp. 707–710, 1966.