LEONARDO RAMOS EMMENDORFER

APRENDIZADO DA LIGAÇÃO ENTRE GENES EM COMPUTAÇÃO EVOLUTIVA: UMA NOVA ABORDAGEM BASEADA EM ESTATÍSTICAS DE BAIXA ORDEM

Tese apresentada como requisito parcial à obtenção do grau de Doutor em Ciências. Programa de Pós-Graduação em Métodos Numéricos em Engenharia, Setor de Ciências Exatas e Setor de Tecnologia, Universidade Federal do Paraná, Brasil.
Orientadora: Profa. Dra. Aurora Trinidad Ramirez Pozo

CURITIBA

2007

SUMÁRIO

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE ABREVIATURAS

ADF funções aditivamente decomponíveis

BB bloco construtor

BOA *Bayesian optimization algorithm*

BSEM *Bayesian Structural Expectation Maximization*

CE computação evolutiva

DCP distribuição condicional de probabilidade

DNF forma normal disjuntiva

EBCOA *evolutionary Bayesian classifier-based optimization algorithm*

EBNA *estimation of Bayesian network algorithm*

ecGA *extended compact genetic algorithm*

EDA algoritmo de estimação de distribuição

GAD grafo acíclico dirigido

hBOA *hierarchical Bayesian optimization algorithm*

HIFF se-e-somente-se hierárquico

KL Kullback-Leibler

LEM learnable evolution model

MIMIC *mutual information for maximization of input clustering*

MLE estimador de máxima verossimilhança

MPM modelo de produto de marginais

PBIL population-based incremental learning

PMBGA algoritmos genéticos baseados na construção de modelos probabilísticos

PV vetor de probabilidade

QV quantização vetorial

RTS *restricted tournament selection*

sGA algoritmo genético simples

UEBNA *unsupervised estimation of Bayesian network algorithm*

UMDA *univariate marginal distribution algorithm*

RESUMO

Este trabalho propõe um novo algoritmo de computação evolutiva, baseado na aquisição de informação a partir das similaridades entre os indivíduos da população e no uso desta informação para guiar a busca. O teorema do esquema, que é uma fundamentação teórica importante da área, já aponta que a existência de similaridades entre os indivíduos está relacionada com a detecção de soluções parciais. Os indivíduos que possuem uma certa sub-estrutura comum podem ser considerados similares, e este grupo de indivíduos é chamado de esquema.

A inferência de modelos estatísticos para representar os melhores indivíduos encontrados é, atualmente, uma abordagem efetiva para computação evolutiva. Modelos cada vez mais complexos vem sendo usados pelos algoritmos de estimação de distribuição (EDAs), o que freqüentemente resulta em uma melhor eficácia. Os EDAs mais efetivos adotam redes Bayesianas, pois estes modelos capturam interações de alta ordem entre as variáveis do problema. O aprendizado da estrutura destes modelos é, entretanto, uma tarefa computacionalmente cara. EDAs baseados em estatísticas de primeira ordem, por sua vez, são computacionalmente mais simples mas não têm se mostrado efetivos em muitos problemas *benchmark*. Isto é devido à incapacidade dos EDAs de baixa ordem em aprender sobre as ligações entre os genes, que são resultantes da estrutura de interações presentes nestes problemas.

O algoritmo proposto é um EDA de baixa ordem que mantém a população agrupada por similaridade e procura explorar o espaço de busca combinando a informação adquirida em grupos diferentes. Uma avaliação empírica é conduzida, a qual adota um escopo abrangente de problemas que representam as principais difuculdades encontradas em EDAs. Os resultados mostram que a nova proposta é efetiva e eficiente em diversos problemas *benchmark*.

ABSTRACT

This work proposes a new evolutionary computation algorithm, which is based on acquiring information from the similarities among the individuals of the population and using this information to guide the search. The schemata theorem, which is an important theoretical foundation for evolutionary computation, already pointed out that the similarities among the individuals is related to the detection of partial solutions. All individuals possessing a certain substructure can be considered similar, and this group of individuals is called schemata.

Inferring statistical models from the best individuals found so far is currently an effective approach for evolutionary computation. Increasingly more complex models have been used by estimation of distribution algorithms (EDAs), which often result better effectiveness. The most effective EDAs adopt Bayesian networks, since those models are able to capture interactions of high order among the variables of a problem. Learning the structure of those models is, however, a computationally expensive task. Simpler EDAs based on single order statistics, on the other hand, are computationally simpler approaches, but have not been shown to be effective on many benchmark problems. This is due to inability of simpler EDAs to learn and respect the linkage among the genes, which is a result of the structure of interactions present in those problems.

The algorithm proposed is a low-order EDA which keeps the population clustered and attempts to explore the space by combining information from different clusters. Empirical evaluation is performed, which adopts a comprehensive range of benchmark problems, which illustrate most of the difficulties found by EDAs. Results show that the new approach is effective and efficient on several *benchmark* problems.

CAPÍTULO 1

INTRODUÇÃO

A computação evolutiva (CE) compreende um conjunto de técnicas de inteligência computacional, inspiradas na teoria de seleção natural de Darwin. A busca pela solução em CE é baseada na representação de populações de indivíduos, que são candidatos a solução, e em mecanismos de seleção, combinação e perturbação, sendo que estes mecanismos podem ser percebidos de forma mais ou menos explícita em cada algoritmo.

A solução de problemas pela computação evolutiva caracteriza-se pelo fato de que nenhuma informação a priori a respeito da estrutura do problema deve, necessariamente, ser incorporada. É necessário um procedimento de avaliação de soluções candidatas, a fim de permitir que o algoritmo seja capaz de aprender sobre o espaço de busca, realizando uma prospecção focada em regiões que se mostram mais promissoras. Este é um conceito chave em CE, o da avaliação da qualidade da solução fornecida por cada indivíduo usando a chamada função de *fitness*. Deve ser possível atribuir um valor que represente o potencial do indivíduo como solução. Em alguns domínios isto pode não ser trivial. Os conceitos de população e geração são também fundamentais para muitos dos algoritmos de CE. A população representa o conjunto de todos os indivíduos que estão sendo considerados candidatos à solução.

As técnicas mais antigas de CE, como o algoritmo genético simples (sGA) [Holland, 1975], expressam de forma bastante explícita os mecanismos de combinação e perturbação, implementando-os na forma de operadores que atuam diretamente sobre a população. Provou-se [Goldberg et al., 1989] que existe uma limitação importante nesta abordagem, que é a necessidade de se incorporar à codificação adotada algum conhecimento a respeito da estrutura do problema. O sucesso do sGA pode depender de uma codificação adequada, variáveis que apresentam interação são codificadas como genes próximos no cromossomo, caso contrário o algoritmo não é capaz de de-

tectar e manter soluções parciais em classes gerais de problemas.

A capacidade de aprender automaticamente sobre a estrutura do problema que está sendo resolvido é chamada de aprendizado de ligação [Harik, 1997]. Percebeu-se que este tipo de aprendizado é necessário para que o algoritmo genético seja capaz de resolver uma classe geral de problemas que apresentam interações entre variáveis [Goldberg et al., 1989]. Problemas *benchmark* vêm sendo propostos na literatura, os quais ilustram situações nas quais o aprendizado de ligação é necessário. Estes problemas a aqui chamada classe de problemas estruturados, em função das estruturas de interação existentes.

Algoritmos evolutivos da classe dos algoritmos de estimação de distribuição (EDAs) [Muhlenbein e Paaβ, 1996] abordam o problema do aprendizado de ligação e da preservação de soluções parciais, por meio de (i) modelagem estatística das variáveis do problema, selecionando uma amostra de soluções promissoras e (ii) geração de novas soluções a partir do modelo estatístico aprendido.

Esta linha de desenvolvimento de EDAs tem se mostrado uma abordagem poderosa para CE. Modelos cada vez mais complexos vêm sendo propostos, o que geralmente resulta em maior eficácia em problemas de otimização estruturados. O estado da arte compreende a adoção de técnicas de aprendizado supervisionado e não supervisionado de redes Bayesianas, como em [Pelikan e Goldberg, 2000] [Pelikan et al., 2005b] [Etxeberria e Larrañaga, 1999] [Larrañaga et al., 1999] [Pelikan, 2005]. A principal propriedade destes modelos é a de tentar capturar informações sobre a estrutura do problema e representar estas informações na forma de um grafo, no qual os nós representam variáveis e os arcos representam dependências. Por meio desta aproximação, interações de alta ordem podem ser capturadas.

Atualmente os EDAs representativos do estado da arte vêm apresentando resultados satisfatórios na solução de problemas que eram considerados difíceis para algoritmos genéticos simples [Pelikan et al., 1999] [Pelikan e Goldberg, 2000]. Entretanto, modelos complexos baseados em redes Bayesianas, requerem uma etapa computaci-

onalmente complexa de aprendizado de estrutura, a qual constitui por si só um problema de otimização NP-completo [Chickering, 1996].

Algumas abordagens [Pelikan e Goldberg, 2000] [Ahn et al., 2004] vêm sendo avaliadas, no sentido fazer com que EDAs mais simples, baseados em estatísticas de primeira ordem apenas, sejam eficazes em um escopo maior de problemas estruturados. Por exemplo, problemas multimodais são abordados por EDAs de primeira ordem paralelizando a busca, de modo que cada subpopulação esteja relacionada a um ótimo específico. Até o momento, entretanto, nenhum EDA de baixa ordem tem se mostrado capaz de resolver problemas estruturados, com presença de interações de alta ordem entre as variáveis do problema.

1.1 Objetivos

O principal objetivo desta tese é o de verificar uma hipótese segundo as similaridades existentes entre indivíduos da população fornecem informações sobre a estrutura do problema. Uma vez capturadas e utilizadas estas informações então um EDA de primeira ordem seria capaz de respeitar a estrutura do problema e manter as interações de alta ordem entre as variáveis.

A proposta se baseia na noção de esquema, como proposto por Holland [Holland, 1975] há mais de trinta anos, para representar a noção de que similaridades entre indivíduos da população são resultado da ocorrência de subestruturas comuns. Esquemas representam regras de especificação parcial de indivíduos [Goldberg, 1998], de modo que um esquema pode incluir mais de um indivíduo. Se o *fitness* dos indivíduos do esquema estiver acima da média, o esquema é chamado de bloco construtor. A combinação entre tais estruturas é um mecanismo fundamental em computação evolutiva e a identificação de blocos construtores deve ser uma preocupação importante [Holland, 1975] [Goldberg, 1998].

Desta forma, um mecanismo baseado na agrupamento de indivíduos por similaridade poderia, teoricamente, capturar informações sobre a estrutura do problema. In-

divíduos que possuem uma certa associação importante de valores para as variáveis, o que constitui uma interação, seriam atribuídos ao mesmo grupo. A combinação de informações sobre grupos diferentes seria capaz de combinar blocos construtores, como sugerido em [Holland, 1975].

Tal abordagem é de interesse prático imediato, pois elimina a necessidade de estatísticas de alta ordem e, principalmente, elimina também necessidade da busca por estruturas de modelos estatísticos complexos. Sendo tão eficaz quanto os algoritmos baseados em redes Bayesianas, a abordagem proposta é mais parcimoniosa e computacionalmente mais eficiente para computação evolutiva.

1.2 Metodologia

A metodologia adotada para verificar a hipótese é a avaliação empírica. Um algoritmo baseado na hipótese é projetado e implementado. Trabalhos anteriores que adotam uma abordagem semelhante em algum sentido são revisados.

A fim de promover uma avaliação empírica adequada, um conjunto representativo de problemas *benchmark* foi selecionado. Os problemas devem incluir as dificuldades mais importantes relatadas na literatura, de modo a se obter um diagnóstico sobre a eficácia do esquema proposto, quando comparado a outros algoritmos competentes de computação evolutiva.

A escolha destes algoritmos competentes, representantes do estado da arte em EDAs também constitui uma etapa necessária. Os EDAs considerados mais competentes, que adotam inferência de redes Bayesianas, são escolhidos para os testes empíricos por serem considerados os mais eficazes atualmente [Chen et al., 2007].

O tipo de experimento também deve ser determinado antecipadamente. Comparações empíricas baseadas na eficácia (capacidade de resolver adequadamente problemas) e eficiência (emprego parcimonioso dos recursos) entre os algoritmos representantes do estado da arte e o algoritmo proposto devem ser realizadas, como meio de validar a hipótese considerada. Outra característica importante a ser avaliada é a es-

calabilidade. É importante que o crescimento no tamanho do problema não implique em aumento exponencial do número de avaliações de *fitness*. A escalabilidade do algoritmo proposto será empiricamente comparada à dos algoritmos representantes do estado da arte.

1.3 Contribuições

As contribuições desta tese são:

- Um ganho em termos de eficiência computacional, relacionado à etapa de inferência da estrutura, que deixa de ser necessária. Ocorre que, até o momento, a resolução de problemas estruturados usando EDAs vem exigindo a inferência de modelos complexos, os quais demandam procedimentos computacionalmente caros para inferência da estrutura.

- Verificação empírica de uma teoria que, embora bastante antiga, não havia ainda motivado um algoritmo tão diretamente baseado nela. Isto porque a hipótese é centrada na noção de esquema [Holland, 1975], que é uma fundamentação teórica muito importante para computação evolutiva.

1.4 Estrutura da tese

Os próximos capítulos estão estruturados da seguinte forma: o capítulo 2 revisa um aspecto central da computação evolutiva, que é o do aprendizado de ligação entre genes. São também apresentados problemas representativos de otimização, que serão adotados na avaliação empírica. Estes problemas foram escolhidos por exigirem um eficiente aprendizado de ligação.

O capítulo 3 discute os principais modelos de inferência que são ou podem ser adotadas em computação evolutiva, a fim de resolver o problema do aprendizado de ligação. Algoritmos existentes que efetivamente adotam estes modelos são revisados, os quais compreendem a área dos algoritmos de estimação de distribuição.

O capítulo 4 apresenta os principais trabalhos relacionados à proposta desta tese. Em linhas gerais, esta revisão inclui as técnicas voltadas à manutenção de diversidade na população, particularmente aquelas baseadas em algoritmos de agrupamento, já que esta abordagem é semelhante à adotada aqui.

O algoritmo proposto é apresentado no capítulo 5. Um mecanismo de combinação de informações obtidas a partir das subpopulações é apresentado, o qual se mostra fundamental no funcionamento do algoritmo.

O capítulo 6 apresenta a avaliação empírica do algoritmo proposto, o que permite discutir a validade da hipótese avaliada. Esta discussão é apresentada no capítulo 7, o qual conclui o trabalho.

CAPÍTULO 2

LIGAÇÃO ENTRE GENES EM COMPUTAÇÃO EVOLUTIVA

Em sistemas biológicos, a ligação entre genes se refere ao nível de associação quando da herança de dois ou mais genes, quando esta é maior que a esperada pela lei Mendeliana da variação independente, que é válida quando dois genes estão em cromossomos diferentes. [Hartl e Jones, 1998]. Quando dois genes estão no mesmo cromossomo, a distância entre eles determina o nível de sua ligação, pois está relacionada à probabilidade de que sejam herdados juntos após o cruzamento, o qual ocorre durante a meiose [Chen et al., 2007]. Na figura 2.1 estão ilustrados dois casos de meiose com cruzamento. No primeiro caso os genes estão próximos, e são herdados juntos na prole. No segundo caso os genes estão distantes e são separados pelo cruzamento.

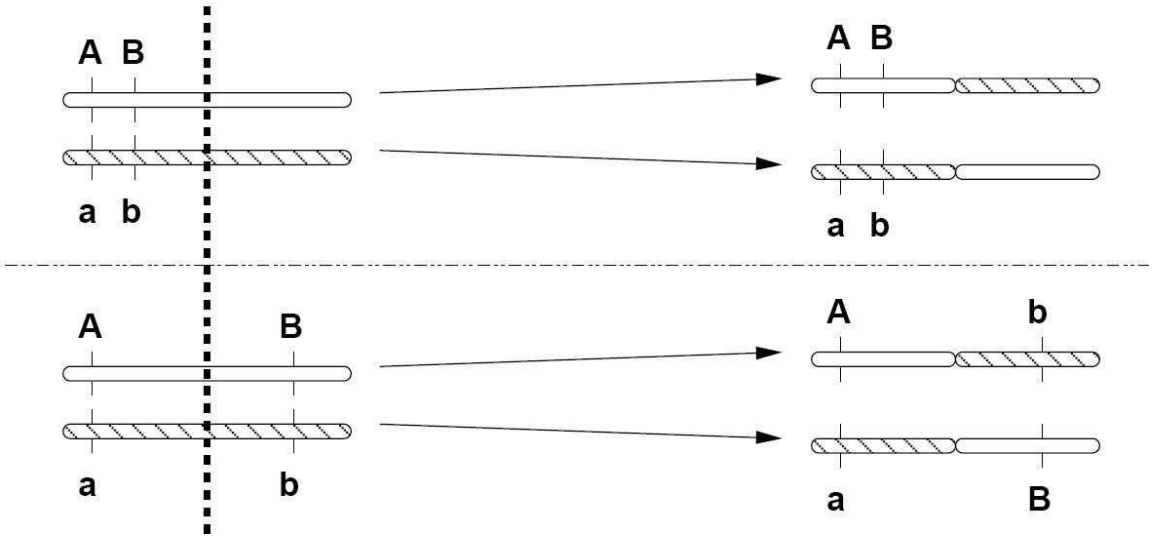


Figura 2.1: Dois exemplos de meiose e cruzamento. A distância entre os genes no cromossomo indica grau de sua ligação (Fonte: [Chen et al., 2007]).

Também em computação evolutiva a interação entre genes deve ser levada em conta [Goldberg et al., 1992]. Isto ocorre porque, na maior parte dos problemas, a contribuição de cada gene para o *fitness* do indivíduo não é independente do valor

dos demais genes.

Este capítulo revisa diversos aspectos sobre o aprendizado da ligação entre genes em computação evolutiva, enfatizando as técnicas adotadas para este fim. Os principais problemas *benchmark*, que ilustram a necessidade deste aprendizado, são revisados ao final do capítulo.

2.1 Computação evolutiva

A computação evolutiva (CE) é um ramo de pesquisa emergente da inteligência computacional que propõe um novo paradigma heurístico para a solução de problemas, inspirado em teorias de evolução. O paradigma original e mais conhecido de CE é baseado na teoria de seleção natural de Darwin, e se originou a partir dos algoritmos genéticos (GAs). O primeiro algoritmo genético [Holland, 1975] atingiu sucesso posteriormente em diversas aplicações, ampliando o interesse por GAs.

O processo evolutivo de um GA pode ser entendido como uma emulação simplificada do processo natural de evolução. Cada candidato à solução é representado por um indivíduo que possui um cromossomo. O conjunto de genes do cromossomo mapeiam atributos do problema, sendo possível usar o mesmo alfabeto do atributo que se quer mapear. Uma abordagem baseada em genes binários também é muito comum. Técnicas de seleção dos melhores indivíduos também estão presentes no algoritmo, sendo que a mais usada é seleção proporcional ao *fitness*. Operadores explícitos de cruzamento e mutação são responsáveis por transformar a população após sucessivas gerações.

A busca pela solução em CE é baseada na representação de populações de indivíduos, que são candidatos à solução, e em mecanismos de seleção, combinação e perturbação, sendo que estes mecanismos podem ser percebidos de forma mais ou menos explícita em cada algoritmo de CE. No sGA estes mecanismos são aplicados como operadores que atuam diretamente sobre os indivíduos da população, enquanto que abordagens mais recentes adotam etapas de inferência sobre a população, procurando

sintetizar toda a informação em um modelo probabilístico.

2.2 Ligação entre genes no algoritmo genético simples

No algoritmo genético simples (sGA) existe um operador específico para combinação, chamado de cruzamento, que efetivamente implementa um processo semelhante ao natural, incorporando no novo indivíduo genes provenientes de cada um de seus pais. Também a perturbação está presente explicitamente no operador de mutação, o qual altera arbitrariamente o valor de um gene.

O sGA é capaz de lidar com a questão da ligação entre genes de forma bastante limitada, desde que exista conhecimento prévio por parte do especialista, sobre quais são as interações existentes entre as variáveis de cada problema. O sGA depende, porém, da ordenação adequada dos genes no cromossomo, o que é feito durante a codificação.

A codificação adequada para o sGA seria a chamada "ligação apertada", na qual as varáveis interagentes devem ser codificadas como genes próximos no cromossomo e o cruzamento deve, mais provavelmente, manter os genes interagentes unidos na prole. O grande problema em relação a esta abordagem é que a obtenção e representação deste conhecimento prévio sobre a estrutura do problema nem sempre são tarefas triviais. Além disso, a existência de interações da ordem do tamanho do problema pode impedir esta abordagem de obter sucesso, já que nestes casos não é possível representar todas as interações como genes próximos no cromossomo. Interações da ordem do tamanho do problema são aquelas nas quais nenhuma variável é independente das demais.

O aprendizado automático, por parte do algoritmo evolutivo, das relações entre variáveis do problema, codificadas como genes, vem sendo reconhecido como uma preocupação importante deste [Goldberg et al., 1989]. Antes, [Holland, 1975] propôs a hipótese de blocos construtores (BBs), a qual discute a preservação e combinação de subestruturas. Operadores genéticos que sejam capazes de aprender informações

sobre qual deve ser a ligação entre os genes podem ser necessários para o sucesso dos algoritmos evolutivos. Esta tarefa vem sendo conhecida como aprendizado de ligação.

2.3 Esquemas e blocos construtores

Os conceitos de esquema e bloco construtor (BB) dizem respeito especificamente à mecânica do funcionamento do algoritmo genético simples. Entretanto, pela sua extensibilidade, estes conceitos são também úteis para explicar outros algoritmos de computação evolutiva, incluindo aqueles mais relevantes para esta tese.

Um esquema é um conjunto de indivíduos que respeitam uma regra. Esquemas representam similaridades entre indivíduos da população [Goldberg, 1998]. A regra de um esquema determina um conjunto de genes especificados e um conjunto de genes não especificados. Tomando um alfabeto binário, a posição s_i de um esquema está associada ao gene i dos indivíduos, de modo que $s_i \in \{0, 1, *\}$, sendo que o símbolo $*$ representa que aquele gene não está especificado no esquema.

Por exemplo, o esquema "11**00" determina uma regra para cadeias binárias de tamanho 6, a qual inclui todos os indivíduos que possuem os dois primeiros genes em 1 e os dois últimos em 0. Os outros dois genes não estão especificados na regra, de modo que apenas quatro indivíduos se incluem neste esquema: "110000", "110100", "111000" e "111100".

Duas propriedades importantes dos esquemas são sua ordem e seu comprimento de definição. A ordem de um esquema é o número de posições especificadas (que são diferentes de $*$). O comprimento de definição é a distância (em posições) entre a primeira e a última posições especificadas no esquema. Estas duas propriedades dizem respeito à forma como o sGA opera sobre esquemas.

A mecânica do sGA é construída, de fato, para operar sobre indivíduos. Porém, esta operação em menor nível produz efeitos significativos na escala mais alta, dos esquemas, a ponto de se afirmar que o sGA opera, de fato, com esquemas. Seja o *fitness* de um esquema definido como o *fitness* médio dos indivíduos pertencentes a este es-

quema. Sabe-se que a seleção, no sGA, escolhe indivíduos com *fitness* mais alto, logo indivíduos membros de esquemas com *fitness* alto serão selecionados mais freqüentemente. Além disso, quanto menor o comprimento de definição do esquema, menor será a probabilidade de que o cruzamento irá romper este esquema. Assim, esquemas com alto *fitness*, comprimento de definição pequeno e baixa ordem possuem grande probabilidade de se proliferarem de geração a geração. Esquemas com estas propriedades são chamados de blocos construtores.

O teorema do esquema afirma que blocos construtores aumentam exponencialmente no tempo, enquanto esquemas de *fitness* abaixo da média decaem numa taxa similar. Isto explica o funcionamento do GA, ao mesmo tempo em que explicita sua limitação em relação à ordem dos blocos construtores.

Uma ressalva importante em relação à aplicabilidade do conceito de blocos construtores foi, mais recentemente, levantada por [Watson, 2002]. Em algoritmos mais recentes de computação evolutiva a ordenação dos genes no cromossomo não é importante, exatamente ao contrário do que ocorre no sGA. O conceito de bloco construtor tal qual entendido por [Holland, 1975] não é relevante para estes algoritmos mais recentes pois não há o cruzamento tal como no sGA. [Watson, 2002, p.28] introduz o conceito mais geral de módulo, que representa um esquema com *fitness* acima da média, independente da sua ordem e de seu tamanho de definição. Nesta tese a expressão "bloco construtor" tem o mesmo significado do módulo definido por [Watson, 2002, p.28], que já é uma prática usual na literatura.

2.4 Técnicas de aprendizado de ligação

O aprendizado automático, por parte do algoritmo evolutivo, a respeito da estrutura do problema que está sendo resolvido tem sido um foco de pesquisa ativo em CE [Chen et al., 2007][Harik, 1997]. Diversas abordagens foram propostas as quais confiam no aprendizado sobre as ligações entre os genes como caminho para o sucesso na resolução de problemas estruturados.

Segundo [Chen et al., 2007], as técnicas existentes de aprendizado de ligação podem ser classificadas em técnicas de ligação física e técnicas de ligação virtual. As técnicas de ligação física são aquelas baseadas na posição de cada gene na cadeia que defina a codificação adotada. Nestas técnicas o aprendizado de ligação emerge a partir da variação das posições de cada gene. Outra categoria inclui as técnicas de aprendizado de ligação virtual. Nestas a ordem física dos genes da codificação é irrelevante: alguma estrutura de dados adicional é adotada para representar informações sobre as ligações entre genes.

2.4.1 Ligação física

Em algoritmos evolutivos baseados em ligação física a ligação entre os genes emerge a partir das localizações físicas dos genes no cromossomo [Chen et al., 2007]. A ligação física é biologicamente mais plausível e é inspirada diretamente na natureza, enquanto que as técnicas de ligação virtual são abordagens de engenharia que procuram atingir o efeito desejado de forma eficaz.

Um exemplo bastante representativo de algoritmo que adota ligação física é o Messy GA [Goldberg et al., 1989]. Neste algoritmo o cromossomo é de tamanho variável, composto de um conjunto de pares (atributo,valor). Esta é uma abordagem baseada em subespecificação e superespecificação, já que um indivíduo pode não ter todos os possíveis genes especificados. No caso da subespecificação, é possível perceber uma aproximação em relação ao conceito de esquema, que é também baseado em uma regra de especificação parcial. A tabela 2.1 ilustra a diferença entre a codificação adotada pelo MessyGA e pelo sGA, e mostra um esquema correspondente ao indivíduo ilustrado.

O MessyGA opera em ciclos de duas etapas: na primeira os blocos construtores são identificados e, subseqüentemente, são combinados entre si, para que se obtenham blocos construtores maiores. Este procedimento é concebido para que as principais sub-estruturas sejam corretamente identificadas antes que a exploração de suas

combinações se inicie.

2.4.2 Ligação virtual

Os algoritmos baseados em ligação virtual procuram capturar e utilizar informações sobre a estrutura do problema, ao invés de tentar alterar a codificação adotada. Embora apresentem menor embasamento biológico, a ligação virtual tem se mostrado mais eficiente do que a ligação física. Uma hipótese para explicar este comportamento, segundo [Chen et al., 2007], seria a de que o real poder dos sistemas biológicos ainda não é bem entendido, logo os algoritmos genéticos ainda não exploram todos os mecanismos críticos necessários existentes em sistemas biológicos.

A partir de uma perspectiva dos métodos de inferência, o aprendizado de ligação está diretamente relacionado à detecção de dependências e interações entre os genes. Como as abordagens de ligação virtual realizam o aprendizado automático sobre a estrutura do problema, então a posição do gene na cadeia passa a ser irrelevante. Algoritmos desta classe incluem os EDAs [Muhlenbein e Paaβ, 1996].

Nos EDAs, os mecanismos de combinação e perturbação são menos explícitos que no sGA; um modelo probabilísico sintetiza a informação sobre os genes provenientes de indivíduos selecionados da população, e a geração de novos indivíduos ocorre a partir da amostragem desta distribuição.

Quando modelos suficientemente expressivos são adotados, então as interações entre as variáveis do problema podem, teoricamente, ser capturadas. Entretanto, para uma classe geral de problemas é necessário capturar interações de ordem alta, muitas vezes da ordem do próprio tamanho do problema. A capacidade de detectar interações

Tabela 2.1: Codificações adotadas no sGA e no MessyGA. Apenas os genes em negrito na codificação GA estão presentes no indivíduo no MessyGA, e no esquema correspondente. Adaptado de [Watson e Pollack, 1999]

0**1**010**01****1**	– sGA
((2,**1**), (6,**0**), (8,**1**))	– MessyGA
1**0***1**	– esquema

de tal ordem exige um número exponencialmente grande de indivíduos na população, o que inviabiliza uma abordagem mais direta, baseada em testar cada uma das possíveis interações de ordem arbitrária.

O capítulo 3 trata, mais detalhadamente, destas abordagens para computação evolutiva que realizam o aprendizado de ligação por meio de modelos baseados em inferência sobre a população.

2.5 Problemas de otimização estruturados

Nesta seção alguns dos problemas de otimização que vêm sendo adotados como *benchmark* para algoritmos de computação evolutiva são revisados, considerando a existência de estruturas de dependência e interação entre variáveis.

A maior parte dos problemas revisados aqui podem ser chamados de problemas estruturados, no sentido em que o *fitness* não depende de cada variável de forma independente, já que existe uma estrutura de interações entre as variáveis do problema. A existência destas estruturas é comum em problemas reais, por isso ela é incorporada nos problemas artificiais adotados na literatura.

Algoritmos mais simples como o sGA e os EDAs de ordem inferior encontram dificuldades em muitos problemas estruturados, pois estes algoritmos não lidam bem com dependências arbitrárias entre variáveis. O sGA com cruzamento de um ponto pode obter bom desempenho mesmo em problemas com interações entre as variáveis, desde que uma codificação adequada seja adotada. Nesta tese, porém, a inclusão de conhecimento prévio sobre a estrutura do problema não é considerada. O paradigma para a resolução de problemas estruturados adotado aqui é o da inferência sobre as interações entre as variáveis do problema, o que está relacionado com o aprendizado de ligação.

Diferentes classes de problemas são abordadas. Elas ilustram as possíveis dificuldades encontradas por diferentes tipos de EDAs, de acordo com os trabalhos que vêm sendo apresentados na literatura de forma a testar a robustez de algoritmos de

computação evolutiva. Um exemplo bastante claro é o dos problemas hierárquicos, nos quais o ótimo global é encontrado após a solução de subproblemas em diversos níveis.

Problemas estruturados se aproximam bastante da descrição feita em [Holland, 1975] na hipótese dos blocos construtores. A existência de subestruturas em problemas de otimização vem sendo abordada pela computação evolutiva praticamente desde suas origens. A existência de blocos construtores está relacionada à ocorrência de subestruturas nos problemas.

Um algoritmo de otimização eficaz deve ser capaz de respeitar estas interações de baixa ordem a fim explorar combinações entre os blocos construtores decorrentes destas interações.

As principais dificuldades apresentadas em problemas *benchmark* podem ser classificadas em: problemas enganosos, hierarquia, sobreposição de subestruturas, multimodalidade global e simetria.

2.5.1 Problemas enganosos: funções armadilha e problemas aditivamente decomponíveis

A dificuldade mais freqüentemente encontrada em problemas *benchmark* é quando um problema é enganoso. Problemas são enganosos quando estatísticas de primeira ordem levam na direção de um bloco sub-ótimo, mas o bloco ótimo correspondente está na direção contrária [Pelikan et al., 1999]. Blocos enganosos correspondem a interações entre os genes. Estatísticas da ordem do tamanho do bloco podem ser necessárias para identificar e manter blocos enganosos.

Funções aditivamente decomponíveis (ADFs) são compostas de uma subestrutura de blocos, possivelmente enganosos, os quais podem ser combinados para obter a solução global. Em ADFs não existem interações entre blocos, no sentido em que a contribuição de cada bloco no *fitness* é independente do valor das demais variáveis. Assim, cada subproblema pode ser resolvido independentemente dos demais, o que

facilita o processo de busca pela solução ótima [Watson et al., 1998].

A seguir é apresentada uma função que ilustra estas características.

2.5.1.1 Função armadilha concatenada

As funções armadilha foram especialmente desenvolvidas para verificar a robustez de algoritmos genéticos. A estrutura do problema definido por uma função armadilha é baseada em grupos de variáveis interagentes, com interações de ordem k. Cada variável do problema está envolvida em uma e apenas uma interação de ordem k, ou seja, cada variável interage com outras $(k-1)$ variáveis. Como os grupos não se sobrepõem e as contribuições de cada grupo para o *fitness* total se somam, estes problemas são chamados de aditivamente decomponíveis. Uma função enganosa determina a contribuição de cada bloco para o *fitness* de um indivíduo. A função armadilha concatenada de ordem 5, ou simplesmente armadilha-5, tem a seguinte forma:

$$trap_5(u) = \begin{cases} 5, & se\ \ u = 5\ , \\ 4-u, & caso\, contrário \end{cases} \tag{2.1}$$

sendo que u é o número de variáveis em 1 no bloco. O *fitness* do indivíduo é obtido pela soma de todas as contribuições. Para cada bloco de tamanho 5 existe um bloco construtor sub-ótimo "00000", com contribuição 4 para a avaliação e um bloco ótimo, com contribuição 5. Na figura 2.2 o caráter enganoso de cada subestrutura está ilustrado, bem como a separabilidade do problema, para uma instância de tamanho 15.

Estatísticas de primeira ordem conduzem ao bloco sub-ótimo "00000". Isto porque blocos de tamanho 5 possuindo poucos 1s tendem, pela forma da função enganosa, a se transformarem gradativamente em blocos do tipo "00000", o que faz com que a quantidade destes blocos cresça na população. Considere uma população inicial aleatoriamente obtida, adotando a proporção binomial $p = 50\%$ para cada gene. Blocos não ótimos como "01010" ou "00101", com distância de Hamming igual a 2 em relação ao sub-ótimo, vão mais provavelmente se tranformar em blocos com distância 1 ao bloco

"00000" do que em blocos com distância 3 ao bloco "11111", pela forma da função 2.1.

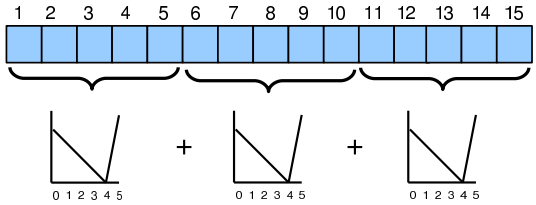


Figura 2.2: Caráter enganoso e separável da função armadilha concatenada de ordem 5.

Múltiplas combinações de blocos ótimos e sub-ótimos levam a diversas soluções ótimas locais. Combinações dos dois blocos construtores "00000" e "11111" em todas as posições da cadeia geram, para partições de tamanho 5 e para um problema de tamanho p um total de $(2^{p/5} - 1)$ ótimos locais e apenas um ótimo global.

Uma instância de tamanho 50, chamada de Ptrapfive50, é considerada nos experimentos, no capítulo 6.

2.5.2 Problemas hierárquicos

Em problemas hierárquicos, existem diferentes níveis de subproblemas a serem resolvidos em ordem. Blocos de menor ordem constituem as soluções de subproblemas do primeiro nível. As soluções do nível subseqüente são obtidas a partir de combinações das soluções de nível inferior. Quando os subproblemas de níveis inferiores vão sendo resolvidos, blocos construtores maiores vão sendo detectados e combinados entre si, até que a solução ótima seja encontrada.

Esta classe de problemas foi proposta após se observar que, em funções ADF, não há interação entre blocos construtores [Watson et al., 1998]. A contribuição de uma subestrutura para o *fitness* do indivíduo não depende do valor das demais subestruturas. Em problemas hierárquicos as subestruturas interagem, fazendo com que a otimização neste tipo de problema seja mais difícil que em funções aditivamente decomponíveis. Teoricamente, em problemas hierárquicos podem existir interações até da ordem do tamanho do problema, o que dificulta a inferência. Entretanto, estas interações podem

ser detectadas a partir das interações de mais baixa ordem por meio de um mecanismo de combinação eficaz.

2.5.2.1 Se-e-somente-se hierárquico

O problema se-e-somente-se hierárquico (HIFF) [Watson et al., 1998] é um exemplo canônico da classe dos problemas hierárquicos. Neste problema existem, por definição, interações da ordem do tamanho do problema. Ele é considerado uma função hierárquica decomponível pois é possível identificar uma estrutura composta de uma hierarquia de subproblemas.

O tamanho da cadeia é determinado pelo número de níveis p na hierarquia. Uma cadeia binária com 2^p bits representa uma solução. O *fitness* de uma solução é dado pela função recursiva:

$$f_{hiff}(\boldsymbol{B}) = \begin{cases} 1,\ se\ |\boldsymbol{B}| = 1 \\ |\boldsymbol{B}| + f(\boldsymbol{B}_L) + f(\boldsymbol{B}_R),\ se\ |\boldsymbol{B}| > 1\ e\ (\forall i\{b_i = 0\}\ ou\ \forall i\{b_i = 1\}) \\ f(\boldsymbol{B}_L) + f(\boldsymbol{B}_R),\ caso\ contrário \end{cases} \tag{2.2}$$

sendo que $\boldsymbol{B}$ é um bloco de bits $(b_1, \ldots, b_{|\boldsymbol{B}|})$, $|\boldsymbol{B}|$ é o tamanho do bloco, b_i é o i-ésimo elemento do bloco. $\boldsymbol{B}_L$ e $\boldsymbol{B}_R$ correspondem às metades esquerda e direita do bloco $\boldsymbol{B}$.

A função recebe como parâmetro um bloco de bits, sendo que a avaliação inicia com a solução inteira como sendo um bloco.

A função premia blocos com bits iguais adjacentes. Os blocos construtores de menor nível são cadeias nas quais ocorrem valores "00" ou "11" iniciando em posições pares (considerando que a cadeia inicia na posição 0). Blocos de níveis maiores correspondem a grupos análogos com tamanhos iguais a potências de dois, até atingir o tamanho do problema, 2^p. O cálculo da função 2.2 está ilustrado na figura 2.3, para a cadeia "00001101". Cada nível da hierarquia considera a avaliação de um nível da es-

trutura em função do nível inferior, a partir da cadeia original. O *fitness* total é a soma da contribuição de todos os nós. Observe que a porção "0000" contribui mais que a porção "1101".

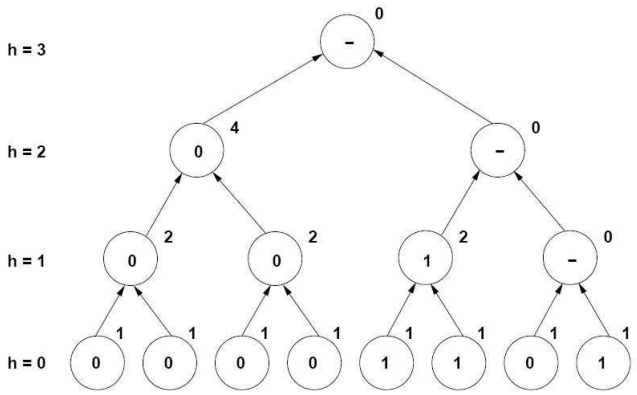


Figura 2.3: Avaliação da função HIFF para a cadeia "00001101". O resultado é a soma de todos os nós.

O problema HIFF original possui a propriedade da codificação apertada, já que genes correlacionados são codificados próximos no cromossomo. Isto faz com que o problema não seja tão difícil para o sGA com cruzamento de um ponto.

A versão do HIFF adotada neste trabalho é o HIFF misturado [Watson et al., 1998], no qual a codificação é baseada numa permutação aleatória dos genes originais do HIFF. Cada rodada implica numa permutação diferente. Isto evita que o sGA tenha sucesso neste problema. Algum mecanismo para detecção da estrutura do problema deve ser aplicado no HIFF misturado. Uma instância do HIFF misturado de tamanho 64 (Pshuff64) é considerada nos experimentos do capítulo 6.

2.5.3 Sobreposição de subestruturas

Uma variação das funções aditivamente decomponíveis é obtida quando se permite que os blocos que definem cada estrutura se sobreponham. Se o *fitness* do problema ainda é definido pela soma das contribuições de cada bloco, mas estes blocos podem compartilhar genes, ou seja, existe sobreposição de blocos, então problemas desta classe são ditos aditivamente decomponíveis com blocos construtores sobrepostos,

conforme [Yu et al., 2005].

O fato de um gene estar presente em mais de um bloco gera dificuldades para a tarefa da inferência da estrutura do problema. Um particionamento das variáveis, por meio de uma fatoração por produtos de marginais, por exemplo, é inviável. Mesmo modelos mais complexos devem, necessariamente, enfrentar maiores dificuldades pois uma estrutura mais complexa de dependência aparece nestes casos.

[Yu et al., 2005] discute como obter uma estratégia ótima de combinação neste tipo de problema, de modo a minimizar o rompimento de blocos construtores.

2.5.3.1 Função armadilha concatenada sobreposta

O problema da função armadilha concatenada sobreposta [Yu et al., 2005] resulta da sobreposição de funções armadilha. É, portanto, um problema aditivo decomponível com blocos construtores sobrepostos. O tamanho da sobreposição de cada bloco com outro é constante; no caso dos problemas verificados aqui este comprimento é 2, com blocos construtores de tamanho 5. Isto significa que cada bloco compartilha 4 genes no total (2 com cada vizinho), com dois outros blocos.

O esquema de sobreposição é circular, como ilustrado na figura 2.4 para uma instância com 6 blocos construtores. Este esquema significa que os dois últimos genes participam do primeiro e do último blocos construtores. Na figura, o bloco 1 compartilha 2 genes com o bloco 6 e 2 genes com o bloco 3.

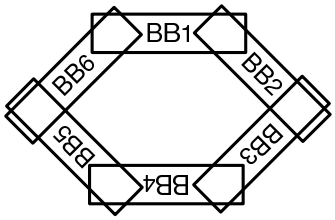


Figura 2.4: Uma instância pequena do problema armadilha-5 concatenada com blocos sobrepostos, com sobreposição de dois genes em cada intersecção.

Uma instância maior, com 60 genes e 20 blocos construtores sobrepostos (também com comprimento de sobreposição 2) chamada Poverfive60 é considerada mais adiante

no capítulo 6. A mesma instância foi também estudada em [Yu et al., 2005].

Na instância considerada aqui, uma permutação dos genes é considerada, de modo que o problema resultante em geral não possui ligação apertada dos genes.

2.5.4 Multimodalidade global e simetria

Problemas nos quais existem diversos ótimos globais são classificados como globalmente multimodais. Problemas de otimização globalmente multimodais representam um cenário de dificuldades para algoritmos genéticos, devido à ocorrência de deslocamento genético (*genetic drift*), que consiste na convergência lenta para apenas um dos ótimos [Goldberg e Richardson, 1987]. A existência de diversos picos faz com que a convergência se torne lenta, já que a combinação de boas soluções vindas de partes diferentes do espaço de busca pode resultar em soluções ruins.

Uma classe de problemas multimodais que vem recebendo atenção da comunidade é a dos problemas simétricos, que apresentam a propriedade de que os ótimos globais ocorrem aos pares, de modo que a cada ótimo global corresponde o seu complementar binário [Pelikan e Goldberg, 2000]. Ou seja, se um indivíduo é ótimo global de um problema simétrico, então seu complementar também o será. A simetria é resultado de regularidades no formato da função de *fitness*, que leva à existência de soluções complementares com *fitness* idêntico e, conseqüentemente, a ótimos globais complementares.

Problemas globalmente multimodais representam dificuldade para computação evolutiva pois a combinação de soluções vindas de ótimos diferentes pode não ser útil para a exploração do espaço de busca, e não contribui para encontrar o ótimo global. Isto é verdade, por exemplo, em problemas reais codificados com variáveis discretas: dois ótimos locais não devem ser combinados nestes casos. Por outro lado, para problemas combinatoriais estruturados a combinação de ótimos locais pode significar a combinação de subestruturas importantes. Este tópico está relacionado diretamente à proposta da tese, e será abordado no capítulo 5.

2.5.4.1 Problema Twomax

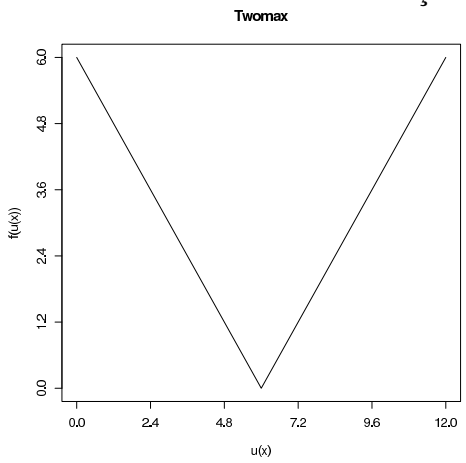
O problema Twomax é definido por uma função simples de n variáveis binárias:

$$f_{twomax}(\mathbf{Z}) = \left| \frac{n}{2} - \sum_{i=1}^{n} z_i \right| \tag{2.3}$$

Existem dois ótimos globais; um é a cadeia "000...0", com $\sum_{i=1}^{n} z_i = 0$ e outro é a cadeia "111...1", com $\sum_{i=1}^{n} z_i = n$. Ambos ótimos apresentam *fitness* igual a $n/2$.

A figura 2.5 ilustra o caráter multimodal da função. O eixo x representa a soma $\sum_{i=1}^{n} z_i$, que corresponde ao número de genes em 1 na solução.

Figura 2.5: Caráter multimodal da função Twomax



Encontrar as duas soluções globais não é trivial, embora a função seja relativamente simples. Teoricamente, existem interações da ordem do tamanho do problema, pois estatísticas de primeira ordem levam a uma indecisão. Como indivíduos próximos de qualquer dos ótimos são igualmente bons, então uma população de bons indivíduos deve conter alguns próximos de "000...0" e outros próximos de "111...1". O modelo estatístico para um gene resulta em uma proporção binomial próxima de $0,5$, que está igualmente distante dos dois ótimos.

Em [Pelikan e Goldberg, 2000] surge uma solução que permite resolver problemas simétricos como este de forma simples usando EDAs de primeira ordem. A população é agrupada, e indivíduos próximos a cada um dos ótimos tendem a permanecer juntos. Assim, modelos estatísticos de primeira ordem para cada grupo convergem rapi-

damente para as respectivas soluções globais.

2.5.5 O problema da bisseção de grafos

O objetivo do problema do particionamento de grafos em duas partes (ou bisseção) é o de dividir o conjunto de nós de um dado grafo em dois conjuntos de igual tamanho, de modo que o número de vértices entre os conjuntos seja mínimo [Peña et al., 2005]. O fitness de uma solução é calculado como o número total de nós menos o número de vértices conectando os dois conjuntos.

A codificação normalmente adotada é baseada em um vetor binário n-dimensional, no qual o i-ésimo gene representa o agrupamento ao qual pertence o i-ésimo nó do grafo. O conjunto de possíveis soluções é aquele em que o número de genes em 1 é igual ao número de genes em 0, já que os conjuntos devem ter tamanhos iguais. É possível perceber que a geração de um novo indivíduo não é uma operação fechada sobre este conjunto de possíveis soluções. Logo, um operador de reparo deve ser adotado, o qual inverte aleatoriamente os valores das variáveis que possuam o valor majoritário (0 ou 1), até que esta diferença esteja corrigida.

O particionamento de grafos ilustra uma característica bastante geral, presente em outros problemas de otimização combinatória, que é a existência de interação, em última instância, entre todas as variáveis do problema, quando a codificação descrita acima é adotada. Para atingir a solução ótima é necessário considerar a relação existente entre todas as variáveis. Esta interação de alta ordem, entretanto, é detectável a partir do aprendizado de interações de ordem menor. Grupos pequenos de nós altamente conectados constituem, para o particionamento de grafos, os blocos construtores iniciais do problema. O algoritmo evolutivo deve ser capaz de, inicialmente, detectar estes grupos de nós e, sucessivamente, explorar a possibilidade de envolver nós vizinhos nestes grupos menores. Ao mesmo tempo, é bastante benéfica, para este problema, a combinação de grupos menores em grupos maiores (coalescência), de acordo a teoria dos blocos construtores, pois este mecanismo implica na sucessiva redução da

dimensionalidade. De acordo com esta discussão, o particionamento de grafos apresenta características semelhantes às dos chamados problemas hierárquicos.

Além de todas estas características, o particionamento de grafos ainda apresenta simetria, já que a inversão de todos os bits de uma solução implica apenas na troca dos rótulos dos grupos (1 e 0) e não acarreta alteração na avaliação do *fitness* da solução.

A maior parte das instâncias consideradas no capítulo 6 está ilustrada nas figuras 2.6 e 2.7; Pgrid16, Pgrid36, Pcat28, Pcatring28, Pcatring42, Pcatring56 e Pcatring84. Outras instâncias com o mesmo prefixo no nome são semelhantes. Por exemplo, a instância Pgrid64 é semelhante topologicamente a Pgrid36, pois é também composta de uma grade regular quadrada de nós, porém em Pgrid64 a grade possui 8 linhas e 8 colunas enquanto Pgrid36 é menor, com 6 por 6 nós. Outras instâncias que não estão ilustradas são Pcat42 e Pcat56, ambas semelhantes a Pcat28. O número de ótimos globais varia de 2 a 6, dependendo da instância considerada. Em função da codificação, como o problema é simétrico, o número de ótimos globais é par, pois a representação de um grupo com 1s ou com 0s é equivalente semanticamente.

Os problemas revisados ilustram as principais dificuldades que são impostas a algoritmos de computação evolutiva. Devido à natureza estruturada destes problemas e à ocorrência de interação entre genes que não estão necessariamente em uma codificação favorável ao sGA, evidencia-se a necessidade de aplicação de alguma técnica de aprendizado de ligação. As principais técnicas foram apresentadas neste capítulo, enquanto que o capítulo 3 discute mais profundamente as técnicas de aprendizado de ligação baseadas em inferência, as quais são consideradas as mais eficazes.

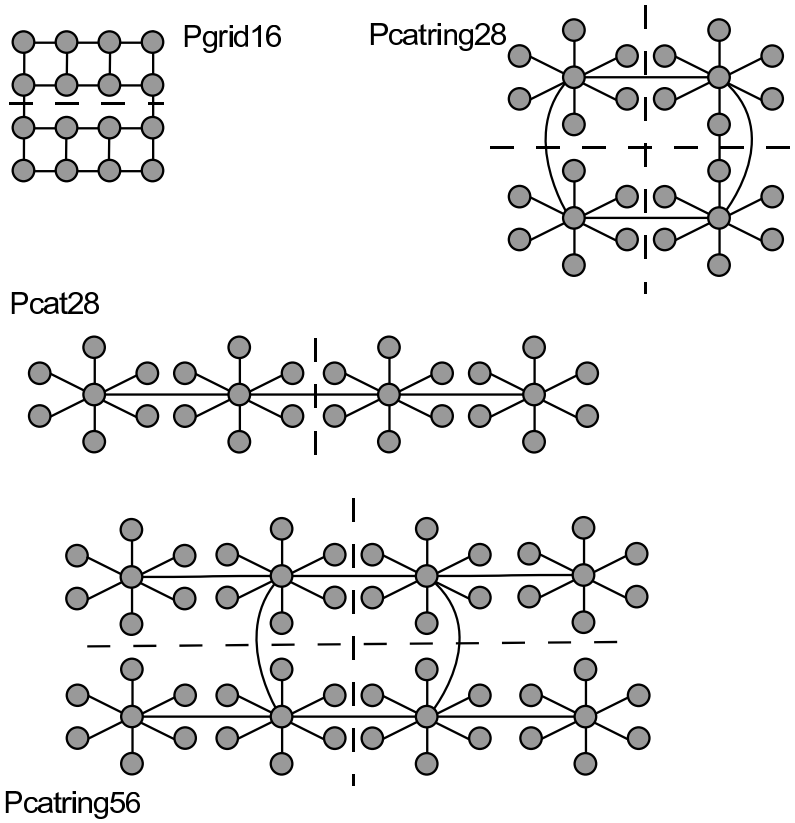


Figura 2.6: Topologias do problema da bisseção de grafos: Pgrid16, Pcatring28, Pcat28 e Pcatring56. Cada linha tracejada ilustra uma possível partição ótima.

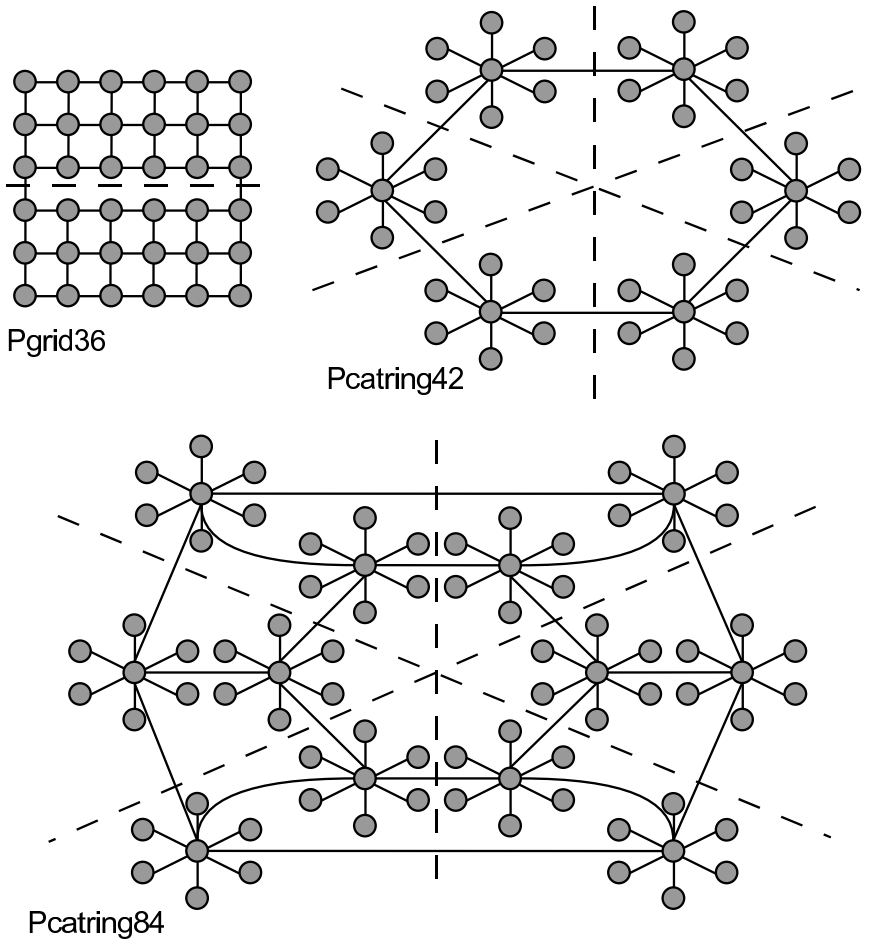


Figura 2.7: Topologias do problema da bisseção de grafos: Pgrid36, Pcatring42 e Pcatring84. Cada linha tracejada ilustra uma possível partição ótima.

CAPÍTULO 3

COMPUTAÇÃO EVOLUTIVA BASEADA EM INFERÊNCIA

Conforme revisado no capítulo 2, o aprendizado sobre a estrutura do problema que está sendo resolvido é uma etapa necessária em computação evolutiva [Goldberg, 1998] [Goldberg et al., 1992]. Sem este aprendizado parece ser impossível resolver problemas estruturados de uma forma escalável, pois a manutenção das subestruturas identificadas permite obter uma redução da dimensionalidade do problema.

Uma possível abordagem para realizar o aprendizado de quais são as relações entre as variáveis do problema é a aplicação de técnicas de inferência para detectar estruturas de dependência e interação entre as variáveis ao longo do processo evolutivo e, assim, procurar preservar estas estruturas de acordo com o modelo aprendido. A abordagem mais usada e que tem obtido mais sucesso é a da inferência de distribuições baseadas em fatoração, que compreende os chamados algoritmos de estimação de distribuição (EDAs) [Muhlenbein e Paaβ, 1996], também chamados de algoritmos genéticos baseados na construção de modelos probabilísticos (PMBGAs). Outras técnicas baseadas em inferência que também foram verificadas na literatura incluem o aprendizado de classificadores, árvores de decisão e indução de regras.

3.1 Inferência e aprendizado de máquina

A área de aprendizado de máquina está relacionada aos métodos computacionais de aprendizado, capazes de adquirir e manter atualizado o conhecimento a partir dos dados e informações disponíveis [Monard e Baranauskas, 2003]. Aprendizado de máquina é uma área da inteligência computacional cujo objetivo é o desenvolvimento de técnicas artificiais de o aprendizado que incluem a construção de sistemas capazes de adquirir conhecimento de forma automática.

Entende-se aqui "conhecimento" como uma forma refinada de informação, enriquecida após processos de inferência e validação [Michalski, 2003]. A inferência é, portanto, um elemento importante da área de aprendizado de máquina. Uma forma de definir inferência é como sendo o processo de calcular probabilidades a posteriori $P(X|Y = Y_0)$ de uma lista X de variáveis de interesse após observar dados $Y = Y_0$. Aqui Y é uma lista de variáveis observadas e Y_0 é a lista de valores observados para estas variáveis [Jakulin e Bratko, 2004].

O paradigma mais importante em aprendizado de máquina é a inferência indutiva (ou aprendizado indutivo), já que este é o fundamento para o aprendizado a partir de exemplos. Um exemplo (ou instância) é uma t-upla (ou vetor) de valores de atributos. Os atributos detalham a informação disponível a respeito dos exemplos [Monard e Baranauskas, 2003]. No aprendizado indutivo, os exemplos são fornecidos a (ou alimentam) um algoritmo, chamado de indutor. O algoritmo indutor deve ser capaz de representar o conhecimento por meio de de alguma estrutura, que pode ou não ser facilmente interpretável por um especialista humano.

Uma abordagem possível para inferência está no conceito de fatoração. Uma fatoração de uma distribuição conjunta de probabilidade é uma lista de fatores (funções) [Zang e Poole, 1999]. Estes fatores permitem expressar relações de dependência e independência entre as variáveis.

Dependências e independências estão relacionadas à existência de interações. Variáveis que interagem são aquelas que fornecem mais informações quando são consideradas em conjunto do que separadamente. Interação representa um todo irredutível; é uma dependência que não pode ser rompida [Jakulin e Bratko, 2004].

A função distribuição conjunta não fatora como um produto de marginais quando existe interação. A interação não é direcional: quando duas variáveis interagem não implica, necessariamente, uma relação de causa e efeito na qual uma é dependente da outra. Alguns modelos de inferência estão limitados à detecção de dependências, então nestes casos as interações existentes serão aproximadas por dependências. Isto

ocorrem em redes Bayesianas, as quais são revisadas a seguir.

Um exemplo clássico da importância das interações em aprendizado estatístico é o problema do ou exclusivo (XOR) [Minsky e Papert, 1969], o qual não é linearmente separável e não pode ser resolvido sem a detecção das interações existentes.

É possível detectar a existência de interações entre variáveis. Pela da divergência de Kullback-Leibler (KL) [Kullback e Leibler, 1951] é possível determinar o quanto a distribuição conjunta se diferencia do produto das marginais. Um teste de hipótese pode ser conduzido, verificando cada uma das possíveis interações de ordens arbitrárias até o tamanho do problema e detectando quais exatamente são as interações entre os genes a partir da população.

Infelizmente, esta abordagem não é tão aplicável na prática [Jakulin e Bratko, 2004] [Zang e Poole, 1999]. Não é possível adotar uma abordagem como esta para ordens superiores de interação, pois o número de possíveis interações cresce em escala combinatória em função do número de variáveis, e seria necessária uma quantidade inaceitável de instâncias para se ter amostras suficientes de todas as interações possíveis.

Fatorações permitem modelar diversos tipos de dependências e interações entre variáveis aleatórias [Pearl, 1988] que expressam se a distribuição de cada variável deve ou não considerar o valor de alguma das demais variáveis. Usando uma fatoração se pretende capturar e respeitar relações importantes entre variáveis, quando se estiver gerando uma amostra da distribuição inferida.

Uma fatoração nem sempre representa perfeitamente a distribuição conjunta. A figura 3.1 representa graficamente as interações pareadas entre um certo grupo de variáveis. As interações mais fortes ocorrem entre as variáveis X_1 e X_4 e entre as variáveis X_2 e X_3. Interações mais fracas acontecem entre outros pares de variáveis. Estas interações mais fracas podem não ser consideradas, por exemplo, em uma fatorização que considere que a distribuição conjunta fatora no produto de duas conjuntas marginais: a conjunta de X_1 e X_4 e a conjunta de X_2 e X_3.

As fatorações são particularmente úteis quando se deseja estimar uma distribuição

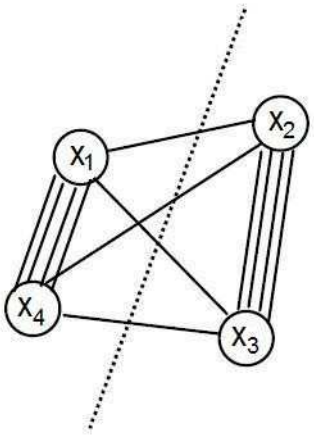


Figura 3.1: Interações entre 4 variáveis.

conjunta a partir de uma amostra multivariada e não se tem conhecimento prévio sobre as estruturas de dependência desta distribuição. Assim, fatorações têm sido importantes em computação evolutiva. A partir da amostra (população), se obtém um modelo de fatoração que permite gerar novos indivíduos, os quais devem respeitar as interações capturadas pela fatoração inferida.

3.2 Algoritmos de estimação de distribuição (EDAs)

Um novo ramo da computação evolutiva tem sido, recentemente, explorado. Os chamados EDAs baseiam-se na adoção de modelos estatísticos que capturam a região (ou regiões) mais promissora(s) do espaço de representação de indivíduos. A principal diferença entre cada EDA é o modelo estatístico assumido *a priori*. Alguns EDAs usam modelos que assumem a independência entre os genes, enquanto outros, mais realisticamente, permitem capturar dependências entre os genes, sendo que a ordem de dependência varia desde apenas pares de genes até ordens superiores. Capturando dependências, os EDAs são capazes de resolver um dos principais problemas do sGA, que é a quebra dos blocos construtores.

Os EDAs caracterizam-se por manter um modelo estatístico das melhores soluções, ou das soluções mais promissoras, encontradas em cada geração, a fim de gerar novas soluções a partir deste modelo. A quantidade de incerteza (aleatoriedade inerente à

variância do modelo) deve, sucessivamente, reduzir-se de modo a possibilitar a convergência após algumas gerações.

A partir de uma população inicial, um EDA seleciona os melhores indivíduos, assim como em outras abordagens de computação evolutiva. A próxima etapa é a de inferência (aprendizado) de um modelo estatístico, a partir destas melhores soluções. Novos indivíduos são gerados a partir do modelo inferido, a seleção é realizada e o processo continua até ocorrer convergência. Este comportamento iterativo dos EDAs está ilustrado na figura 3.2.

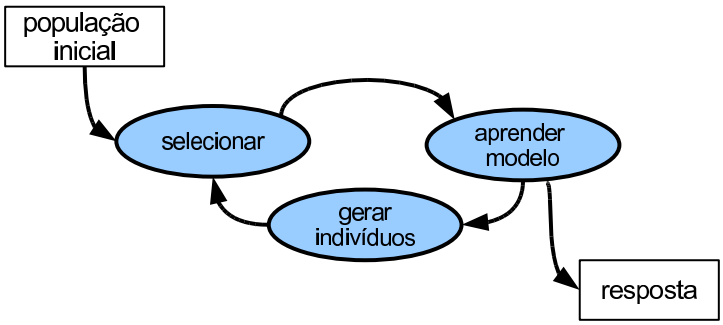


Figura 3.2: Ciclo de funcionamento de um EDA

Da perspectiva de inferência, o conjunto de soluções promissoras pode ser visto como uma amostra tomada a partir de uma distribuição desconhecida [Pelikan et al., 2002]. A distribuição ideal seria uniforme para cada gene, tendo a própria solução ótima, única, com p=1. Entretanto, a partir da execução de um EDA, a informação obtida pode permitir construir apenas uma distribuição mais vaga, carregando alguma variância associada a viés e incerteza. Porém, esta distribuição poderia representar adequadamente as regiões mais promissoras do espaço de busca.

O algoritmo 1 é uma descrição em alto nível do processo adotado em EDAs.

Muitos dos EDAs foram propostos, inicialmente, para aplicações de domínio discreto, e neste trabalho o foco estará restrito exatamente a este tipo de aplicação. Assim, nesta tese assume-se que os genes mapeiam variáveis discretas com alfabeto binário.

Apesar das diferenças, o viés imposto na busca por EDAs e por GAs é semelhante [Pelikan et al., 2002]. Este viés favorece soluções que podem ser obtidas por meio da

Algoritmo 1 Pseudocódigo de um EDA

```
Gerar uma população inicial aleatoriamente.
Calcular o fitness de cada indivíduo.
enquanto critérios de convergência não forem satisfeitos faça
    Construir um modelo probabilístico dos melhores indivíduos.
    Amostrar a partir deste modelo para gerar novos indivíduos e atualizar a
    população
fim enquanto
```

combinação parcial de soluções encontradas. A diferença reside na forma como esta combinação é realizada. Estimar uma única distribuição para representar a diversidade de soluções encontradas pode teoricamente capturar a estrutura do problema e garantir uma mistura efetiva e a reprodução das subestruturas relevantes [Pelikan et al., 2002]. Entretanto, estimar a estrutura completa, capturando todas as interações necessárias, não é uma tarefa simples; existe um compromisso entre precisão e eficiência da estimativa que é muito particular nestes algoritmos.

A combinação de informações é realizada em EDAs, já que um único modelo é obtido a partir diversos bons indivíduos. Infelizmente, combinar indivíduos diferentes pode levar a resultados muito ruins se o modelo adotado não for suficientemente expressivo. Quando o modelo é expressivo, ele é capaz de capturar as dependências, as quais deveriam representar e respeitar as interações existentes no problema. Assim, a motivação para inferir modelos mais complexos e poderosos é a de obter informações sobre a estrutura do problema.

EDAs mais simples confiam em modelos estatísticos de primeira ordem. Isto significa que se assume independência entre os genes. São modelos computacionalmente econômicos, principalmente porque não existe a etapa de busca pela estrutura do modelo, já que esta é fixa [Harik et al., 1999]. Além disso, a concepção simples e a facilidade de implementação os fazem modelos muito atraentes. Entretanto, sua pouca eficácia em problemas considerados difíceis para algoritmos genéticos não é aceitável. Isto constitui uma grande desvantagem, já que a computação evolutiva vem sendo conhecida pela robustez e ampla aplicabilidade [Peña et al., 2005].

Por outro lado outros EDAs procuram realizar o aprendizado sobre as ligações entre os genes inferindo modelos probabilísticos expressivos, os quais geralmente envolvem a busca por uma fatoração adequada. Modelos nos quais interações entre duas ou mais variáveis são consideradas adotam estatísticas de ordem maior que um. Embora bons resultados venham sendo relatados, a desvantagem destes métodos reside exatamente na necessidade de realizar esta busca por estrutura que envolve, geralmente, a aplicação de algum algoritmo guloso que investiga dentro de um espaço combinatório de possíveis estruturas. Um custo computacional muito alto está associado a essa busca pela fatoração ótima, e o resultado sub-ótimo nem sempre é satisfatório [Pelikan et al., 2005a] [Yuan e Gallagher, 2005].

Nas próximas seções são apresentados os principais modelos adotados pelos EDAs. Os modelos são apresentados em ordem de complexidade, a partir do modelo mais simples, que assume independência entre as variáveis, até os modeos mais complexos, capazes de capturar interações de mais alta ordem. Exemplos de EDAs que adotam cada um dos modelos são também apresentados.

3.2.1 EDAs baseados em modelos de primeira ordem

Nesta classe de EDAs, assume-se um modelo de independência entre os genes, logo a fatoração do modelo é trivial. As proporções dos valores de cada gene são inferidas independentemente. O modelo é dito de primeira ordem pois considera cada variável individualmente, enquanto modelo de segunda ordem incluem também estatísticas sobre interações entre variáveis.

Esta fatoração é a mais simples, e está relacionada aqui para fins ilustrativos. Em um modelo que assume independência entre as variáveis, a distribuição conjunta fatora no produto das distribuições marginais de cada variável:

$$\boldsymbol{\rho}(\boldsymbol{x}) = \prod_{i=1}^{p} \rho_i(x_i) \tag{3.1}$$

Tabela 3.1: Tabela verdade da função XOR, para $X_3 = X_1$ XOR X_2.

X_1	X_2	X_3
0	0	0
0	1	1
1	0	1
1	1	0

sendo que $\boldsymbol{\rho}$ é o modelo conjunto,

$\boldsymbol{x}$ é uma instância, com valores para todas as variáveis;

$\rho_i(x_i)$ é o modelo marginal para a variável X_i

Quando as variáveis são binárias, os modelos $\rho_i(x_i)$ representam proporções binomiais, nas quais a fração de sucessos é interpretada como o número de indivíduos com o valor 1 no gene i.

O modelo 3.1 não considera eventuais interações existente, pois se assume que as variáveis sejam independentes. Para ilustrar a deficiência na expressividade deste tipo modelo tomemos o problema do XOR.

A tabela 3.1 representa a função $X_3 = X_1 XOR X_2$. Dada apenas a tabela e buscando inferir um modelo para a distribuição conjunta das variáveis de acordo com o modelo de independência, obtêm-se modelos independentes para cada variável:

$$\pi_1(X_1) = \pi_2(X_2) = \pi_3(X_3) = 0,5 \tag{3.2}$$

sendo que $\pi_i(X_i) = P(X_i = 1)$, ou seja, a probabilidade de que a variável assuma o valor 1.

Este modelo não é capaz capturar as informações mais relevantes contidas nos dados, já que cada variável tem probabilidade igual a $1/2$ tanto para assumir o valor 0 quanto para assumir o valor 1. Quando a interação entre as variáveis X_1 e X_3 for considerada, então um modelo mais adequado pode ser obtido.

Assumindo um mapeamento em variáveis binárias, existem p parâmetros a serem estimados, sendo que p é o número de genes na codificação. Cada parâmetro corres-

ponde à proporção de indivíduos que possuem o valor 1 em um dado gene. O conjunto de parâmetros é, freqüentemente, referido como vetor de probabilidade (PV), ou também vetor de proporções.

3.2.1.1 Algoritmos PBIL e cGA

O primeiro representante desta abordagem é o PBIL (*Population-Based Incremental Learning*) [Baluja e Caruana, 1995]. O PBIL adota uma estratégia de aprendizado incremental competitivo, no qual o vetor representa inicialmente a incerteza máxima em relação a cada um dos genes. Realizando passos de aprendizagem competitiva, novos indivíduos são gerados e competem. Incrementalmente, o vetor é sempre deslocado na direção do indivíduo vencedor.

Outros algoritmos também se baseiam na estrutura de independência, entre eles o cGA (*Compact Genetic Algorithm*) [Harik et al., 1999]. Este algoritmo é construído de modo a garantir a mesma solução que o sGA com cruzamento uniforme, porém o cGA é mais eficiente pois não precisa representar toda a população explicitamente.

Assim, embora estes EDAs processem as soluções de uma forma diferente em relação ao sGA, já foi teoricamente provado e empiricamente verificado que os resultados podem ser semelhantes [Harik et al., 1999]. O sGA com cruzamento uniforme seleciona aleatoriamente um valor de cada posição do vetor um dos pais. Assintoticamente, isto equivale a obter amostras a partir das distribuições obtidas para cada gene.

3.2.2 EDAs baseados em modelos de segunda ordem

Outras classes de EDAs procuram detectar interações entre os genes. Estatísticas de segunda ordem permitem inferir relações pareadas entre genes.

Na fatoração de segunda ordem cada variável depende apenas de uma outra no conjunto. Um exemplo deste tipo de fatoração é apresentado em [Larrañaga et al., 1999], com o nome de fatoração em árvore:

$$\boldsymbol{\rho}(\boldsymbol{x}) = \prod_{i=1}^{p} \rho_i(x_i|x_{\pi_i}) \tag{3.3}$$

sendo que $\boldsymbol{\rho}$ é o modelo conjunto,
$\boldsymbol{x}$ é uma instância, com valores para todas as variáveis;
$\rho_i(x_i)$ é o modelo marginal para a variável X_i;
X_{π_i} é a variável da qual X_i depende.

Pelo menos uma das variáveis deve ser considerada como sendo independente de todas as das demais; esta ocupa a raiz da árvore. Neste caso, π_i é vazio. As variáveis dos próximos níveis são consideradas diretamente dependentes da variável que está no nível imediatamente anterior e, indiretamente, dependentes de todas as variáveis nos níveis superiores ao que ela está. Em cada nível existem os nós de decisão, que representam os valores para a variável considerada. As subárvores inferiores a um nó de decisão representam, então, distribuições condicionadas ao valor dado por aquele nó e por todos os outros nós superiores no caminho até a raiz. Os últimos nós da árvore são chamados folhas e originam distribuições de probabilidade condicionadas, sendo que as condições são dadas pelo caminho percorrido da raiz até cada folha.

3.2.2.1 Algoritmo MIMIC

Um exemplo desta abordagem é o MIMIC - *Mutual Information for Maximization of Input Clustering* [Bonet et al., 1997]. Este algoritmo assume que o modelo que descreve a região promissora do espaço de busca é como em (3.3), com dependências pareadas entre as variáveis.

Ou seja, no MIMIC assume-se uma estrutura de dependência de forma encadeada, com um gene independente (X_{i_p}) e com cada um dos demais genes sempre dependentes de apenas um outro na cadeia. A cada geração, é necessário identificar inicialmente a estrutura do modelo, o que se resume à determinação da permutação de genes que minimizam a divergência de Kullback-Leibler (KL) entre o modelo (3.3) e um modelo

ideal, que fosse capaz de detectar todas as possíveis dependências. A permutação que gera o modelo com menor divergência KL em relação ao modelo ideal é escolhida. Resta apenas a determinação dos parâmetros do modelo, o que é feito estimando todas as condicionais a partir da população.

Por exemplo, para um modelo com duas variáveis X_1 e X_2, suponha que o modelo escolhido seja o de independência de X_2 e dependência de X_1 em relação a X_2. Então os parâmetros que devem ser estimados são apenas três: $\phi_1 = P(X_2 = 1)$, $\phi_2 = P(X_1 = 1|X_2 = 0)$ e $\phi_3 = P(X_1 = 1|X_2 = 1)$, respeitando a estrutura de dependência escolhida e assumindo um alfabeto binário.

3.2.3 EDAs baseados em produtos de marginais

Os EDAs estudados até aqui apresentam limitações na estrutura de dependências existente entre as variáveis. Entretanto, muitas problemas reais envolvem a necessidade de capturar graus maiores de dependência, apresentando estruturas menos restritivas que as abordadas pelos algoritmos estudados até este ponto.

Modelo de produtos de marginais (MPMs) são fatorações bastante simples. Neste tipo de modelo se assume que as variáveis interagem em partições, ou grupos mutuamente exclusivos. Apenas duas hipóteses são plausíveis para cada variável em um modelo de produtos de marginais: ou a variável é independente de todas as demais (neste caso está em uma partição de tamanho 1), ou ela faz parte de alguma das partições – grupos nos quais ocorre interação. Uma variável pode participar de apenas uma das partições.

Dentro de cada grupo não há fatoração; para realizar inferência é necessário considerar todas as possíveis combinações de valores das variáveis discretas envolvidas. Considere um exemplo no qual existem 5 variáveis binárias e se deseja inferir um MPM no qual existem dois grupos: $g_1 = \{X_1, X_3, X_4\}$ e $g_2 = \{X_2, X_5\}$. Os grupos são considerados independentes. O número de parâmetros que deverão ser inferidos para o modelo depende do tamanho dos grupos. Considerando que a distribuição marginal

de cada grupo possui $(2^{k_i} - 1)$ parâmetros, sendo que k_i é o tamanho do i-ésimo grupo, então o grupo 1, composto de 3 variáveis binárias, possui $(2^3 - 1 = 7)$ parâmetros, enquanto o grupo 2 possui 3 parâmetros. O total de parâmetros do exemplo é a soma destes, ou seja, 10 parâmetros.

A fatoração obtida por um MPM para k grupos é:

$$\boldsymbol{\rho}(g_1, g_2, \ldots, g_k) = \prod_{i=1}^{k} \rho_i(g_i), \quad g_1 \cup g_2 \cup \cdots \cup g_k = \{x_1, x_2, \ldots, x_p\} \tag{3.4}$$

sendo que $\boldsymbol{\rho}$ é o modelo conjunto

$g_1, g_2, \ldots, g_k$ são os grupos de variáveis nas quais o vetor $\boldsymbol{X}$ fatora;

$\{x_1, x_2, \ldots, x_p\}$ é uma instância, com valores para todas as variáveis;

$\rho_i(x_i)$ é o modelo marginal para a variável X_i;

X_{π_i} é a variável da qual X_i depende.

Percebe-se que em um MPM o tamanho dos grupos não pode ser muito grande pois o impacto sobre a complexidade do modelo é considerável. Outras fatorações, revisadas a seguir, são capazes de aproximar interações de alta ordem usando modelos menos complexos considerando dependências entre variáveis.

Para o exemplo da figura 3.1, uma fatoração adequada por MPM seria a ilustrada na figura 3.3. A fatoração representada ali é:

$$\boldsymbol{\rho}(X_1, X_2, X_3, X_4) = \rho_{1,4}(X_1, X_4)\rho_{2,3}(X_2, X_3) \tag{3.5}$$

Ou seja, apenas as interações mais fortes, entre X_1 e X_3 e entre X_2 e X_4 foram consideradas.

3.2.3.1 Algoritmo eCGA

O eCGA – algoritmo genético compacto estendido – [Harik, 1999] é baseado na escolha de uma fatoração adequada por produtos de marginais. A busca pela melhor fatoracão

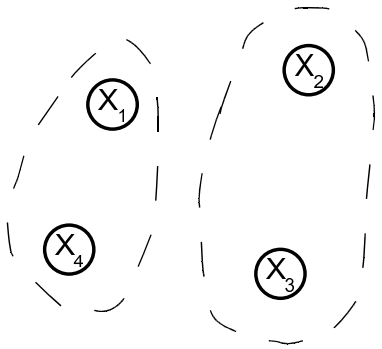


Figura 3.3: Uma fatoração por MPM para as variáveis mostradas na figura 3.1.

é feita usando a métrica de comprimento mínimo de descrição, o qual penaliza tanto modelos imprecisos quanto modelos complexos, assim fazendo com que a distribuição se aproxime da ótima. De acordo com este princípio, a melhor distribuição é aquela que promove a melhor compressão, considerando a codificação associada à representação da população sob aquela distribuição. Formalmente, a complexidade de um modelo MPM é dada pela soma da complexidade do próprio modelo C_m mais a complexidade da população comprimida sob este modelo, C_p [Lima et al., 2005]. A complexidade C_m do modelo quantifica a representação do modelo em termos do número de bits necessários para representar todos os parâmetros, que são probabilidades marginais. Seja um problema de tamanho l com codificação binária, com m partições, com k_i genes em cada partição i, tal que $\sum_{i=1}^{m} k_i = l$. Então cada partição requer $(2^{k_i} - 1)$ freqüências independentes para definir sua distribuição marginal. Considerando que cada freqüência tem tamanho $log_2(n+1)$, sendo que n é o tamanho da população, então a complexidade do modelo C_m é dada por:

$$C_m = \log_2(n+1) \sum_{i=1}^{m} (2^{k_i} - 1). \tag{3.6}$$

A complexidade da população comprimida, C_p, que quantifica a compressão em termos da entropia da distribuição marginal sobre todas as partições, é dada por:

$$C_p = n \sum_{i=1}^{m} \sum_{j=1}^{2^{k_i}} -p_{ij} log_2(p_{ij}), \tag{3.7}$$

sendo que p_{ij} é a freqüência da j-ésima seqüência de genes dos genes que pertencem à i-ésima partição.

O eCGA executa uma busca gulosa por partições a cada geração. Esta busca começa com o modelo mais simples que assume que todas as variáveis são independentes e segue tentando unir partições enquanto o critério de comprimento mínimo é melhorado.

3.2.4 EDAs baseados em redes Bayesianas

Os EDAs considerados mais competentes atualmente adotam modelos mais expressivos, baseados em redes Bayesianas, que são capazes de representar relações de ordem maior que dois entre os genes, sem a necessidade de expressar todas as condicionais completas dentro de cada grupo de genes interagentes, como acontece em modelos de produtos de marginais. Com redes Bayesianas, dependências entre os genes podem ser capturadas, resolvendo potencialmente o problema do aprendizado de ligação.

Uma rede Bayesiana [Pearl, 1988] é um modelo probabilístico baseado em grafos usado para representar uma fatoração para um conjunto $\boldsymbol{X}$ de variáveis aleatórias discretas, $\boldsymbol{X} = \{X_1, X_2, \ldots, X_p\}$, admitindo-se que existem relações de dependência entre as variáveis. Redes Bayesianas são usadas para representar dados multivariados em vários domínios de aplicação, sendo úteis em situações em nas quais a estrutura de dependência entre as variáveis é desconhecida ou apenas parcialmente conhecida *a priori*. Supondo um conjunto de p variáveis aleatórias $X_1, X_2, \ldots, X_p$, então a partir de um conjunto de amostras multivariadas pode ser possível construir uma rede Bayesiana que seja capaz de representar (i) quais são os relacionamentos de dependência entre as variáveis e (ii) a distribuição conjunta destas variáveis, assumindo estes relacionamentos de dependência.

O número total de possíveis dependências é função combinatória do número de variáveis, de modo que detectar e levar em conta apenas as dependências que sejam mais importantes é essencial para obter um modelo expressivo, sem a necessidade de um número grande de amostras.

A rede Bayesiana deve ser capaz de representar estas relações de dependência por meio de um modelo como:

$$\boldsymbol{\rho}(\boldsymbol{x}) = \prod_{i=1}^{p} \rho_i(x_i|\boldsymbol{pa}_i) \tag{3.8}$$

sendo que $\boldsymbol{\rho}$ é o modelo conjunto,

$\boldsymbol{x}$ é uma instância, com valores para todas as variáveis;

$\rho_i(x_i)$ é o modelo marginal para a variável X_i;

$\boldsymbol{pa}_i$ é o conjunto de pais da variável X_i, ou seja, das quais X_i depende.

Fica claro nesta fatoração que as variáveis são consideradas condicionalmente independentes. Para cada variável X_i existe um conjunto de "pais", que inclui as variáveis das quais X_i é considerada dependente. Cada variável, dados seus pais, é considerada independente das demais. Encontrar a melhor fatoração para uma dada amostra é, em si, um problema de otimização bastante complexo [Etxeberria e Larrañaga, 1999], o que força a adoção de alguma heurística. Esta fatoração é possível pois, de acordo com a independência condicional [Pearl, 1988] duas variáveis X_a e X_b podem ser consideradas independentes, condicionado ao conhecimento do valor de outras k variáveis, X_{v_1}, X_{v_2},... X_{v_k}.

Uma Rede Bayesiana possui dois componentes. O primeiro é um grafo acíclico dirigido (GAD), sendo que cada vértice corresponde a uma variável aleatória. O GAD descreve a estrutura de dependência de uma distribuição. Uma aresta saindo de X_i para X_j indica que X_j depende de X_i. O outro componente inclui uma distribuição condicional de probabilidade (DCP) para cada variável X_i, as quais complementam a informação do grafo e descrevem a probabilidade condicional de cada variável, dados seus pais. Por exemplo, se a variável X_i possui k pais (k arestas incidentes) então a DCP completa de X_i possui 2^k parâmetros, um para cada combinação de valores dos pais (considerando um alfabeto binário).

Para a estrutura da figura 3.1 existem diversas redes Bayesianas possíveis. A figura 3.4 ilustra seis possíveis estruturas de rede Bayesiana que podem ser obtidas a partir de uma estrutura como a da figura 3.1. Talvez as melhores sejam aquelas que consideram as dependências relacionadas às duas interações mais fortes. A fatoração (b), por exemplo, capturou também uma interação fraca entre X_3 e X_1, além das duas interações mais fortes, entre X_1 e X_4 e entre X_2 e X_3. O modelo para a rede da figura 3.4b é:

$$\boldsymbol{\rho}(\boldsymbol{x}) = \rho_4(x_4)\rho_2(x_2)\rho_3(x_3|x_2)\rho_1(x_1|x_3, x_4) \tag{3.9}$$

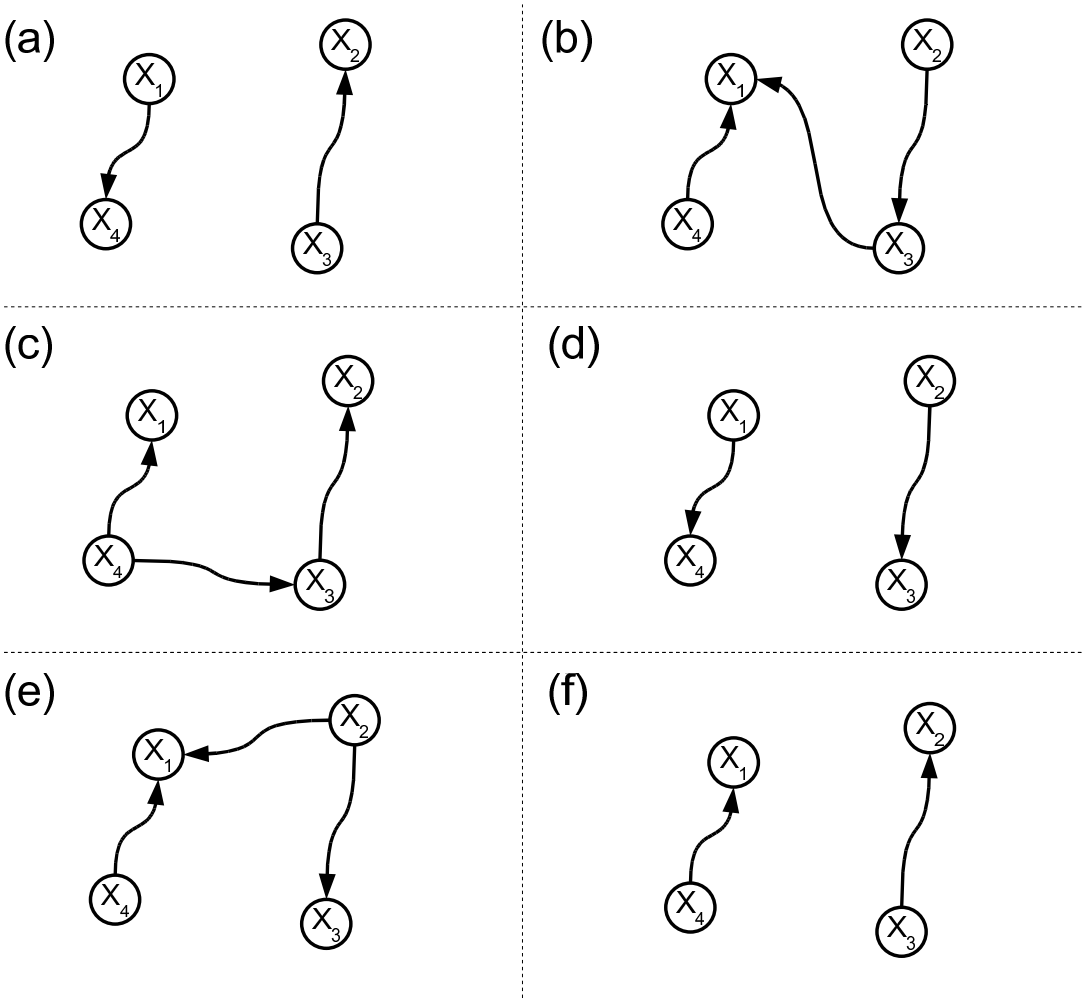


Figura 3.4: Seis das possíveis fatorações por rede Bayesiana para as variáveis mostradas na figura 3.1.

O conjunto de DCPs de uma Rede Bayesiana permite descrever completamente o

modelo dado em (3.8). Logo, uma vez obtida a estrutura da rede, então basta descrever cada DCP para obter um modelo completo para a amostragem a partir da rede. A construção de DCPs a partir de uma tabela de dados é trivial já que, conhecidos os valores dos pais, a distribuição de cada variável é multinomial.

O algoritmo K2 [Cooper e Herskovits, 1992] é um dos empregados para o aprendizado da estrutura da rede. Este algoritmo adota uma método heurístico guloso para a busca entre as possíveis estruturas. O objetivo é maximizar alguma medida de qualidade da rede. Partindo de uma grafo sem arestas, a cada etapa a estrutura é alterada (arestas são inseridas), e avaliada. Uma possível medida de qualidade de rede é a métrica Dirichlet Bayesiana, que representa a probabilidade conjunta de (i) o conjunto de dados e (ii) a estrutura de rede que está sendo avaliada. É feita uma restrição, em termos de quais relações ancestrais são aceitas da seguinte forma: uma ordenação ancestral deve ser respeitada, de modo que uma variável só pode ser pai das variáveis que estão em uma posição posterior, e não pode ser pai de nenhuma variável que tenha posição anterior à sua.

Uma vez obtida a estrutura, então todas as DCPs podem ser calculadas diretamente a partir dos dados. Simular, a partir do modelo inferido para (3.8), é também bastante direto, respeitando a ordenação das variáveis. A amostragem inicia da variável menos dependente (que é, na verdade, independente de todas as demais) seguindo a ordem até a variável mais dependente.

3.2.4.1 Algoritmos BOA e EBNA

O BOA – *Bayesian optimization algorithm* – [Pelikan et al., 1999] e o EBNA – *estimation of Bayesian network algorithm* [Etxeberria e Larrañaga, 1999] são EDAs que adotam um modelo do tipo rede Bayesiana para representar a população das melhores soluções a cada geração.

BOA e EBNA têm sido aplicados a problemas de otimização de diversas classes. Uma destas classes é a de problemas decomponíveis: nestes casos, existe uma estru-

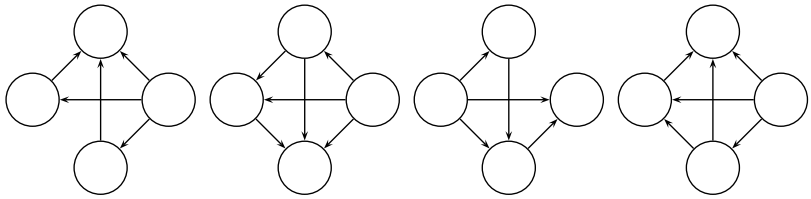


Figura 3.5: Estrutura de uma rede Bayesiana aprendida durante o processo evolutivo de um EDA para o problema armadilha concatenada de ordem 4

tura de interação entre as variáveis problema, de modo que é possível ao algoritmo identificar adequadamente os subproblemas envolvidos.

É importante perceber a forma como a rede Bayesiana representa as interações existentes no problema. Tomemos uma função armadilha de ordem 4 [Pelikan et al., 1999]. As interações são aproximadas como dependências entre as variáveis. Neste caso, não faz muito sentido a direção da dependência, mas por restrição do modelo estas direções estão presentes. Este caso está ilustrado na figura 3.5. É possível notar que nem todas as possíveis dependências são sempre detectadas. A estrutura de cada subproblema é representada por um subgrafo específico, os quais variam em relação à quantidade de dependências detectadas.

3.2.4.2 Algoritmo hBOA

O algoritmo hBOA – *Hierarquical Bayesian Optimization Algorithm* [Pelikan, 2005] é baseado em três princípios: (i) a capacidade de decompor o problema em subproblemas menores capturando os blocos construtores; (ii) a capacidade de capturar a estrutura hierárquica a partir dos níveis inferiores até uma ordem qualquer; (iii) a capacidade de preservar soluções alternativas, evitando convergência prematura. A decomposição do problema pode ser resolvida por meio de modelos tipo redes Bayesianas.

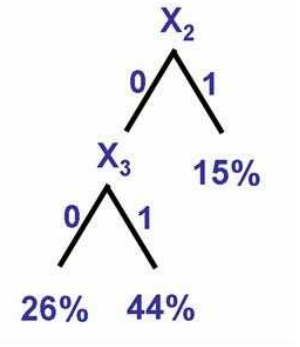
Entretanto, para representar adequadamente a hierarquia do problema, é necessário capturar interações de ordem proporcional ao tamanho do problema. Isto leva a um potencial crescimento exponencial do número de interações, conseqüentemente invia-

bilizando a solução computacional. Entretanto, a fim de contornar este problema, um modelo baseado em estruturas locais em Redes Bayesianas foi adotado. Neste tipo de estrutura, uma árvore de decisão T_i (ou grafo de decisão) é aprendida para cada variável X_i, de modo a representar a distribuição de probabilidade daquela variável, condicionada às demais. As variáveis presentes na árvore T_i poderão ser consideradas "pais" da variável correspondente X_i.

Duas possíveis formas de representar a DCP de uma variável, usando uma tabela ou uma árvore de decisão, estão ilustradas na figura 3.6. A adoção de estruturas locais é fundamentada por um argumento prático, já que o número de instâncias de alta qualidade cresce num limite bem menor que o exponencial (possivelmente, linear), permitindo a construção de árvores expressivas, mesmo com populações relativamente pequenas.

X_2X_3	$P(X_1=0\|X_2X_3)$
00	26 %
01	44 %
10	15 %
11	15 %

(a)



(b)

Figura 3.6: Duas formas de representar uma DCP para a variável X_1 (a) uma enumeração completa em uma tabela e (b) uma árvore de decisão. Adaptado de [Pelikan, 2005].

A manutenção de soluções alternativas ao longo da execução do hBOA é garantida pelo método RTS (*Restricted Tournament Selection*) [Harik, 1995]: um indivíduo novo vai substituir aquele que for mais parecido com ele na população em termos de algum critério de similaridade. O critério mais usado nestes casos é a distância de Hamming.

3.2.5 Algoritmos baseados em classificação e aprendizado de regras

Alguns algoritmos procuram realizar inferência de modelos de classificação, que permitem obter regras que separam os grupos de indivíduos de *fitness* superior dos indivíduos de *fitness* inferior. A idéia fundamental é a indução de classificadores, ou de conjuntos de regras, como sendo o mecanismo fundamental que guia o processo evolutivo. O trabalho pioneiro é o chamado algoritmo LEM [Michalski, 2000] ou *Learnable Evolution Model*. São selecionados indivíduos com *fitness* extremo, os quais dão origem a dois grupos: P_+ e P_-, representando os de maior e menor *fitness*, respectivamente. A partir destes grupos são aprendidas as regras simbólicas capazes de diferenciar indivíduos destas duas classes. Um trabalho semelhante é o algoritmo SI3E [Llorà e Goldberg, 2003]. As linhas gerais da proposta são semelhantes às do LEM, sendo que o algoritmo de aprendizado indutivo usado foi o ID3, e as regras que classificam indivíduos derivam da árvore de decisão obtida a cada geração.

Um outro trabalho que segue uma linha semelhante é [Miquélez et al., 2004], no qual são adotados classificadores na forma de Redes Bayesianas, aplicando-os de forma semelhante ao LEM e SI3E. No *evolutionary Bayesian classifier-based optimization algorithm* (EBCOA), entretanto, são criadas várias classes representando a discretização do *fitness*dos indivíduos em mais de dois níveis. Os autores defendem que esta abordagem evita a perda de informação, já que cada nível de *fitness* seria explicado por seu próprio conjunto de regras, embora nenhum experimento seja realizado no sentido de verificar esta afirmação.

Os EDAs revisados neste capítulo ilustram o funcionamento desta classe de algoritmos. A adoção de modelos baseados em estatísticas de alta ordem vem sendo a abordagem mais eficaz, já que este tipo de modelo permite capturar a estrutura de problemas considerados difíceis, como os revisados no capítulo 2. Entretento, EDAs de alta ordem exigem alto custo computacional na etapa de aprendizado da estrutura do problema que está sendo resolvido. Motivadas neste aspecto, algumas abordagens vêm sendo propostas no sentido de melhorar a eficácia de EDAs de baixa ordem sem impor um

grande aumento no custo computacional, através do agrupamento da população. O trabalho aqui apresentado se insere neste contexto. Trabalhos relacionados a este por compartilharem objetivos semelhantes são revisados no capítulo 4.

CAPÍTULO 4

TRABALHOS RELACIONADOS

Neste capítulo as principais abordagens relacionadas a proposta desta tese são revisadas. De um modo geral, as técnicas que segmentam a população de EDAs e paralelizam a busca são abordadas. Entre estas técnicas se incluem a aplicação de manutenção de diversidade em EDAs e, mais especificamente, a adoção de algoritmos de agrupamento em EDAs.

4.1 Manutenção de diversidade em GAs e EDAs

Um aspecto muito importante em GAs é o da preservação da diversidade da população. Segundo Sastry [Sastry et al., 2005] os mecanismos de manutenção de diversidade devem ser capazes de manter a diversidade da população ao longo do processo evolutivo, permitindo aos EDAs que: (1) identifiquem múltiplos ótimos de forma confiável ao resolver problemas multimodais e multiobjetivos (2) identifiquem o ótimo global após decidir com sucesso entre subestruturas quando interações hierárquicas existirem (3) identifiquem rapidamente soluções globais assim que ocorrerem mudanças, no caso de problemas não-estacionários. Além disso, o mecanismo de manutenção de diversidade também apóia a identificação de subestruturas no problema, já que uma subestrutura só pode ser identificada se houver, na população, um número suficientes de indivíduos que a possuam.

As técnicas de manutenção de diversidade foram introduzidas em GAs, inicialmente, para prevenir a convergência prematura a ótimo locais. Para capturar a estrutura de problemas estruturados, independente da metodologia adotada para este fim e independente do número de ótimos globais existentes, a preservação da diversidade da população revela-se um mecanismo crucial [Mahfoud, 1995]. Somente mais tarde

estas técnicas foram ampliadas e estendidas para permitir a manutenção estável de múltiplos ótimos globais [Mahfoud, 1995].

As técnicas de manutenção de diversidade existentes, aplicáveis tanto em EDAs quanto em outros algoritmos de computação evolutiva, podem ser classificadas em dois tipos [Mahfoud, 1995]: (i) baseadas em compartilhamento de recursos e (ii) baseadas em subpopulações geograficamente separadas. Ambas abordagens são revisadas a seguir.

4.1.1 Diversidade por compartilhamento de recursos

Nas abordagem por compartilhamento de recursos, indivíduos semelhantes devem competir entre si pelos recursos reprodutivos existentes. Isto implica em uma vantagem competitiva inicial para indivíduos que diferem significativamente dos demais, pois eles não competem tanto quanto os indivíduos que são mais comuns. A pressão de seleção se encarrega de eliminar indivíduos que, apesar de diferenciados, não apresentam vantagens em termos de *fitness*.

Um dos primeiros estudos neste sentido está em [Cavicchio, 1970], por meio de um mecanismo chamado de pré-seleção. Nesta abordagem, um filho substitui o pai com menor *fitness*. A preservação de diversidade está garantida de certa forma, pois indivíduos semelhantes entre si estão competindo por um lugar na população.

O trabalho de De Jong [Jong, 1975], chamado de *Crowding*, é uma generalização da pré-seleção. No *Crowding*, um indivíduo é comparado a uma subpopulação selecionada aleatoriamente. O indivíduo mais parecido na subpopulação é substituído. Uma variação é o *Crowding* determinístico [Mahfoud, 1995], que se baseia no mesmo princípio, porém a regra de aceitação é determinística.

Outra forma de preservar diversidade é pelo compartilhamento de *fitness* [Goldberg e Richardson, 1987]. No compartilhamento, uma função que define a degradação do fitness de cada indivíduo é adotada. O fitness de um indivíduo é reduzido, em função do número de indivíduos semelhantes a ele que existirem na população. Isto

impõe um viés de preservação de diversidade, já que indivíduos com subestruturas pouco freqüentes apresentam vantagem competitiva.

O limiar até o qual um indivíduo é considerado semelhante ao outro é dado pelo chamado raio de compartilhamento. Se a distância entre dois indivíduos é menor do que o raios de compartilhamento, eles são considerados semelhantes. Ajustar o raio de compartilhamento não é trivial, e este parâmetro é muito relevante para a eficácia do mecanismo de manutenção de diversidade.

Em [Petrowski, 1997] o método *Clearing* é proposto, o qual se baseia na separação da população em subpopulações e no elitismo dentro de cada subpopulação. O indivíduo de maior fitness em cada subpopulação é considerado o indivíduo dominante, e todo o potencial reprodutivo é atribuído apenas para estes indivíduos dominantes. Outros trabalhos, como [Li et al., 2002] também seguem uma linha parecida, mantendo forte elitismo dentro de cada subpopulação. Uma linha diferente é seguida por [Sastry et al., 2005], que propõe que diversidade seja mantida explicitamente no nível dos blocos construtores aprendidos por um EDA.

Alguns mecanismos de manutenção de diversidade têm sido mais usados em EDAs. Um deles é a seleção restrita por torneio (RTS) [Harik, 1995]. Na RTS, a substituição de indivíduos atuais por novos segue um critério de similaridade: o indivíduo atual será substituído por um novo, que seja o mais parecido com o indivíduo atual. Além disso, é imposta a condição de que o fitness seja maior, já que o RTS é, originalmente, um método de seleção.

4.1.2 Diversidade por separação geográfica

Uma alternativa ao compartilhamento de recursos é a manutenção de diversidade por imposição de estrutura geográfica na população. Os indivíduos são mantidos em grupos, ou subpopulações, e evoluem de modo aproximadamente paralelo. Podem ou não haver migrações entre subpopulações mas, de um modo geral, estas migrações são controladas.

Algoritmos evolutivos que adotam esta abordagem são chamados de algoritmos espacialmente estruturados.

No contexto de EDAs, um trabalho que aplica diretamente este conceito é o PBIL paralelo, ou P^2BIL [Ahn et al., 2004]. Múltiplas subpopulações são mantidas, e um vetor de probabilidade representa o modelo estatístico para cada subpopulação. É proposto um operador simples de combinação entre vetores chamado cruzamento uniforme de vetores de probabilidade (PVs) o qual é semelhante ao cruzamento no algoritmo genético simples. Dois PVs são combinados selecionando-se aleatoriamente de qual PV será copiada cada proporção binomial. Um novo indivíduo é gerado a partir do PV combinado.

Uma abordagem relacionada à separação geográfica é a manutenção de diversidade por algoritmos de agrupamento [Pelikan e Goldberg, 2000][Peña et al., 2005], ou por aprendizado não supervisionado. Estas técnicas vêm sendo avaliadas recentemente, e são revisadas na próxima seção.

4.2 Algoritmos de agrupamento em EDAs

Foi verificado que a aplicação de algoritmos de agrupamento (ou aprendizado não supervisionado) melhoram o desempenho de EDAs para certas classes de problemas. No caso de problemas globalmente multimodais, por exemplo, algoritmos de agrupamento são particularmente atraentes [Peña et al., 2005]. Tais problemas apresentam múltiplos ótimos globais, ou seja, múltiplas soluções globais com o mesmo *fitness*. Para o sGA, sabe-se que esta propriedade resulta em convergência muito lenta para apenas um dos ótimos, conseqüência da combinação não efetiva entre indivíduos próximos a ótimos diferentes [Mahfoud, 1995]. Comportamentos semelhantes ocorrem em EDAs. A manutenção de diversidade por meio de algoritmos de agrupamento vem se mostrando efetiva no sentido de evitar este tipo de problema [Pelikan e Goldberg, 2000] [Peña et al., 2005].

Isto ocorre pois estes problemas exigem mecanismos poderosos de manutenção de

diversidade para que sejam resolvidos [Mahfoud, 1995] e para que todos os ótimos globais sejam encontrados. Ocorre que grande parte dos problemas reais são multimodais. A multimodalidade global decorre também da codificação: problemas de otimização combinatória, como o particionamento de grafos, estão sujeitos à existência de soluções equivalentes semanticamente, logo com fitness idêntico, decorrentes da inversão de cada bit da representação binária adotada, já que a rotulagem de cada grupo, no caso da bisseção de grafos por exemplo, com 0s ou com 1s é intercambiável. Este tipo de multimodalidade global é conhecida como simetria, conforme revisado no capítulo 2.

4.2.1 Agrupamento em EDAs de baixa ordem

A primeira vez que algoritmos de agrupamento foram aplicados para manutenção de diversidade em EDAs foi em [Pelikan e Goldberg, 2000]. Neste trabalho, foi aplicado um EDA univariado, que assume independência entre os genes, a problemas multimodais simétricos, adotando o algoritmo de agrupamento particional *k-means* [McQueen, 1967] como mecanismo de manutenção de diversidade, agrupando, por similaridade de genótipo, os indivíduos.

O resultado mais notável é a grande melhoria da efetividade e da eficiência do UMDA – univariate marginal distribution algorithm, que é um EDA de primeira ordem – quando é aplicado o agrupamento. Neste caso, o número de ótimos globais encontrados e mantidos de forma estável aumenta consideravelmente. Os resultados se explicam pela especialização de cada grupo em um ótimo diferente. Entretanto, apenas problemas de estrutura muito simples como o Twomax foram resolvidos com o UMDA agrupado.

Para problemas mais estruturados, foi verificado que o esquema adotado não melhora o desempenho do UMDA. Embora seja muito desejável que EDAs simples, de baixa ordem, sejam capazes de resolver problemas complexos, até recentemente não havia nenhuma proposta eficaz neste sentido.

4.2.2 Agrupamento em EDAs de alta ordem: algoritmo UEBNA

Um outro trabalho que também aplica agrupamento em EDAs é o algoritmo de estimação não supervisionada de redes Bayesianas (UEBNA) [Peña et al., 2005]. Um linha diferente é seguida, pois a intenção não é a de promover um incremento de desempenho de um EDA de baixa ordem, mas sim validar um novo EDA de alta ordem, que inclui ainda mais elementos na sua proposta, utilizando problemas *benchmark* conhecidos.

Este trabalho adota uma abordagem para agrupamentos baseada em modelos, no sentido em que o rótulo do grupo é considerado como sendo também uma variável aleatória. Este rótulo entra no modelo de inferência como uma variável C, sendo que para esta variável não há valores nas instâncias.

Uma rede Bayesiana representa o modelo probabilístico fatorado para os melhores indivíduos. Na fatoração, são incluídos todos os genes e mais a variável de grupos C. Espera-se que o algoritmo seja capaz de detectar a atribuição correta dos ótimos globais a subpopulações diferentes, ao mesmo tempo em que captura as dependências entre genes por meio do restante da rede.

O modelo adotado pelo UEBNA é ilustrado na figura 4.1a. Nem todas as variáveis são dependentes de C; as relações existentes devem ser inferidas. Como não há valores fornecidos previamente para C, existe uma componente não-supervisionada de aprendizado.

O algoritmo de aprendizado não supervisionado de redes Bayesianas adotado pelo UEBNA é o *Bayesian Structural Expectation Maximization* (BSEM) [Friedman et al., 1998]. Este algoritmo se baseia numa busca gulosa pela melhor estrutura, inferindo a cada etapa valores para os rótulos desconhecidos.

A validação empírica confirma a efetividade do algoritmo para um conjunto de problemas multimodais estruturados. O problema mais relevante testado foi a bisseção de grafos, que é reconhecido como um problema difícil mesmo para EDAs de alta ordem como o BOA e o EBNA. Os resultados mostram que UEBNA supera o EBNA, atingindo um número maior de ótimos globais, especialmente para as instâncias maiores do par-

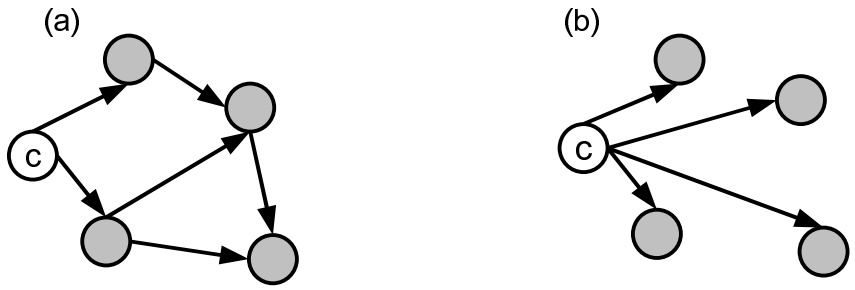


Figura 4.1: (a) Modelo de fatoração adotado pelo UEBNA [Peña et al., 2005], o qual inclui um rótulo de grupo na estrutura e exige, por isso, o aprendizado não-supervisionado de redes Bayesianas (b) o modelo adotado implicitamente por EDAs de primeira ordem paralelos como o P^2BIL [Ahn et al., 2004] ou quando sujeitos a agrupamento [Pelikan e Goldberg, 2000]. Assume-se independência entre todas as variáveis, e a dependência existe apenas entre cada variável e o rótulo de grupos C. Este modelo é adotado explicitamente pelo algoritmo φ-PBIL proposto.

ticionamento de grafos, o que confirma a relevância do aprendizado de agrupamentos no contexto de computação evolutiva.

Infelizmente, modelos complexos baseados em redes Bayesianas, tais como os adotados por EBNA, UEBNA, BOA e hBOA requerem uma etapa computacionalmente complexa de aprendizado de estrutura, a qual deve ser realizada novamente a cada geração. Encontrar a melhor estrutura constitui um problema de otimização muito difícil. Foi provado analiticamente [Chickering, 1996] que a identificação da estrutura da rede entre todas as possíveis, limitando o número de pais de cada nó a um inteiro $t > 1$ é um problema de otimização NP-completo. Na prática, algoritmos gulosos acabam sendo utilizados, o que leva a resultados sub-ótimos.

4.3 O problema do crescimento da ordem de interação

A combinação entre BBs de baixa ordem deve levar a BCs de ordem maior, o que aumenta o grau de dificuldade na detecção e manutenção destes blocos. Dito de outra forma, algoritmos evolutivos devem ser capazes lidar com interações de ordens sucessivamente maiores ao longo do processo. Esta característica está presente com clareza nos problemas hierárquicos. À medida que se avança no processo evolutivo, para estes problemas, o tamanho das subestruturas aumenta claramente, como ilustrado no

capítulo 2.

Sabe-se que o AG simples lida bem com este problema, desde que uma codificação adequada seja adotada, conforme discutido no capítulo 2. Como não se pode confiar no conhecimento prévio da melhor codificação, então esta vantagem do AG é discutível para um contexto geral de aplicação.

EDAs de alta ordem, baseados em redes Bayesianas, também podem encontrar dificuldades ao tentar buscar dependências entre as variáveis quando a ordem de interação cresce até o tamanho do problema, já que as interação existentes são detectadas aproximadamente pelas dependências entre pares de variáveis. Quando a dimensionalidade da interação for próxima de 2, é razoável aceitar a aproximação por dependências. Para interações maiores esta aproximação pode não ser tão efetiva.

Este pode ser chamado de problema da ordem crescente de interação. Uma parte da rede detecta a interação correspondente a um bloco construtor A, enquanto uma outra parte detecta o bloco B. Ao longo do processo evolutivo, a detecção da combinação das estruturas $A \cup B$ exige que a rede Bayesiana seja capaz de capturar uma interação de ordem maior, o que implica numa crescente dificuldade. A detecção do novo bloco construtor é bastante custosa: uma nova busca por uma fatoração adequada deve ser realizada, tentando encontrar dependências entre todas as variáveis envolvidas no novo bloco $A \cup B$. À medida em que o bloco fica maior, torna-se mais difícil encontrar uma fatoração adequada, já que dependências de alta ordem tornam-se mais importantes. Isto representa uma limitação mesmo para os EDAs de alta ordem, que pode acontecer mesmo em problemas simples, como mostrado em [Coffin e Smith, 2007].

Os trabalhos revisados neste capítulo aplicam se baseiam em aplicar aprendizado não supervisionado sobre a população de um EDA, com o objetivo de manter esta população separada em grupos, de modo que a prevenir a combinação entre estes grupos.

A aplicação de agrupamento com este objetivo corresponde ao oposto do que motiva a hipótese aqui avaliada. Ocorre que, nos trabalhos revisados neste capítulo, se

espera que os algoritmos de agrupamento auxiliem a separar indivíduos que estejam associados a ótimos globais diferentes, que não podem ser combinados. No capítulo 5 é apresentado um EDA onde o interesse na aplicação dos agrupamentos é outro: o de detectar interações entre as variáveis, as quais revelam informações sobre a estrutura do problema.

CAPÍTULO 5

ALGORITMO PROPOSTO

Esta tese avalia a hipótese de que um mecanismo baseado em agrupar indivíduos por similaridade poderia capturar informações sobre a estrutura do problema. A noção intuitiva que embasa esta hipótese encontra raízes na idéia de esquemas, que são similaridades entre indivíduos da população [Holland, 1975]. Segundo a hipótese que está sendo verificada, um algoritmo de agrupamento poderia capturar estas informações, as quais podem ser combinadas no sentido de se explorar adequadamente o espaço de busca.

Nos trabalhos existentes na literatura e revisados no capítulo 4, a aplicacão de técnicas de agrupamento em EDAs vem sendo motivada pela intenção de ajudar o algoritmo a separar partes complementares do espaço de soluções. Segundo [Pelikan e Goldberg, 2000], "evitando a combinação de soluções vindas de grupos diferentes, o efeito negativo da simetria em um problema pode ser aliviado". A intenção é melhorar a eficácia dos EDAs de primeira ordem em problemas multimodais, porém os resultados não são satisfatórios. A simples paralelização da busca por meio da separação do espaço em regiões isoladas não se mostrou uma abordagem eficaz para detectar a estrutura de problemas complexos, como o particionamento de grafos por exemplo. A intenção nestes trabalhos é a de agrupar indivíduos para evitar a combinação entre grupos diferentes, enquanto que a hipótese verificada neste trabalho é a de que esta combinação entre grupos diferentes deve, sim, ser realizada.

Assim, para avaliar a hipótese, deve ser proposto um EDA baseado em estatísticas de primeira ordem cuja população seja mantida agrupada e no qual a combinação entre grupos seja propiciada por um mecanismo específico. O algoritmo estará restrito a problemas com variáveis binárias. A validação é empírica, feita a partir de um conjunto de problemas *benchmark* da literatura. Esta avaliação procura comparar os resultados do

novo algoritmo com os dos representantes do estado da arte em computação evolutiva.

5.1 Inferência de agrupamentos

A detecção de grupos é conhecida como agrupamento. As técnicas de agrupamento permitem identificar similaridades entre os exemplos, detectando a existência de grupos mais homogêneos, de acordo com os valores dos atributos. Uma família importante de métodos de agrupamento, dentro do contexto do aprendizado de máquina, está relacionada à quantização vetorial (QV) de Kohonen.

As linhas gerais da QV são semelhantes aos métodos estatísticos de agrupamento não-hierárquico [Bacao et al., 2005]; particularmente como o algoritmo *k-means* [McQueen, 1967]. O agrupamento pode ser definido como um problema de inferência estatística, considerando que as instâncias são amostras provenientes de populações distintas; o método k-médias deve ser capaz de obter os centros de grupo, de modo a maximizar a confiança de que eles pertençam a populações diferentes.

O objetivo da QV é obter os parâmetros que descrevem cada um dos grupos, sendo que cada parâmetro representa o valor esperado de um atributo para aquele grupo. O chamado "centro" do grupo é a média dos vetores de atributos para todos os membros do grupo(instâncias). Então deve-se minimizar a variabilidade dentro de cada grupo, ao mesmo tempo que maximizar a variabilidade entre os grupos. Da mesma forma que nas k-médias, o processo pode é semelhante a um problema inverso de análise de variância (ANOVA), de modo que os centros são escolhidos de modo a obter uma ANOVA com máxima significância em um teste de diferença entre médias.

O algoritmo *k-means* [McQueen, 1967] opera com filosofia semelhante. Dado um conjunto $\boldsymbol{X}$ com N pontos, o k-médias separa $\boldsymbol{X}$ em k grupos com aproximadamente a mesma variância. O algoritmo procede atualizando o conjunto de k centróides, sendo que cada um deles está associado a um grupo. Os centróides podem ser inicializados aleatoriamente, mas algoritmos mais recentes podem ser usados para encontrar atribuições iniciais melhores. Cada iteração consiste de dois passos: no primeiro, cada

ponto é associado ao centro mais próximo (empates podem ser resolvidos arbitrariamente). No segundo passo os centróides são recalculados de modo que cada um deles é análogo a um "centro de massa" dos pontos associados a ele, num espaço de dimensão igual ao número de variáveis. O algoritmo termina quando todos os pontos em $\boldsymbol{X}$ permanecem no mesmo grupo depois de recalcular os centróides e reassinalar os pontos aos novos centróides. Os pontos associados a cada centróide definem o grupo. Pode ser necessário rodar o algoritmo mais de uma vez para diferentes sementes de números aleatórios para obter resultados satisfatórios [Pelikan et al., 2005b].

5.2 Detecção de esquemas e blocos construtores por meio do agrupamento

A idéia de aplicar agrupamentos para que um EDA de primeira ordem seja capaz de capturar blocos construtores é bastante direta: espera-se que indivíduos no mesmo esquema sejam automaticamente agrupados juntos pelo algoritmo. A seleção fará com que apenas bons indivíduos permaneçam; logo, os esquemas existentes estarão, sucessivamente, representando blocos construtores.

A existência de subestruturas no problema é um importante fator que explica a diversidade da população. Quando o algoritmo está nos estágios iniciais, relativamente longe da convergência, a principal fonte de diversidade é em função da variedade de subestruturas locais, já que ainda não se encontrou nenhum ótimo global. Assim, nestes estágios do processo, um algoritmo de agrupamento deve ser capaz de manter juntos indivíduos que possuam as mesmas subestruturas, ou seja, pertencentes ao mesmo esquema.

Os melhores indivíduos devem ser, incrementalmente ou a cada geração, agrupados por semelhança de genótipo. Cada grupo define uma subpopulação. Modelos independentes baseados em proporções binomiais são inferidos para cada gene.

Esta fatoração se baseia em identificar grupos de instâncias na base, cada grupo sendo caracterizado pela ocorrência de uma interação importante. Esta abordagem

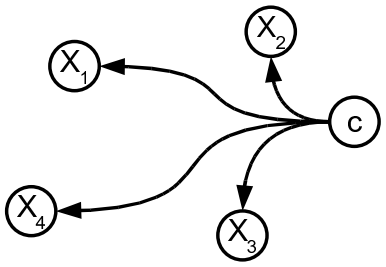


Figura 5.1: Fatoração por agrupamentos.

parte do princípio de que, embora a quantidade total de interações seja exponencialmente grande, o número de interações que são realmente importantes é limitado e estas interações, mesmo que de alta ordem, podem ser detectadas por agrupamento.

O modelo de fatoração para esta abordagem está ilustrado na figura 5.1. É um modelo bastante parcimonioso, com uma estrutura fixa de dependência. A base de instâncias é agrupada e uma variável c é adicionada, a qual representa o rótulo de grupo. Cada variável original é dependente apenas do rótulo de grupo. Assim, o modelo resultante está limitado apenas a estatísticas de ordem 2.

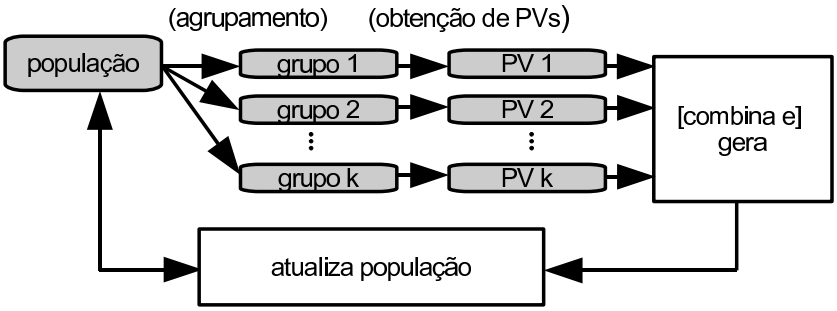


Figura 5.2: Arquitetura do algoritmo φ-PBIL e de outros EDAs de primeira ordem com agrupamentos

Assumindo um número fixo de k grupos e um número fixo de p genes, então um total de $p \cdot k$ proporções binomiais independentes devem ser mantidas atualizadas ao longo do processo evolutivo. Denota-se aqui por $\hat{\pi}_{i,j}$ a proporção de indivíduos do grupo i que possuem o gene j com valor 1. A matriz $\hat{\Pi}$ armazena estas proporções Binomiais. Novos indivíduos devem ser gerados a partir do modelo representado por $\hat{\Pi}$, amostrando-se a partir de algum dos vetores de probabilidade, os quais correspondem às linhas de $\hat{\Pi}$. A combinação de vetores é considerada aqui, embora não seja muito

comum na literatura, conforme revisado no capítulo 4.

A arquitetura do algoritmo proposto é ilustrada na figura 5.2. O esquema é muito parecido com outras abordagens existentes [Pelikan e Goldberg, 2000] [Ahn et al., 2004]. A etapa "combina" representa a possibilidade de combinação de informações pelo cruzamento uniforme de vetores de probabilidade de subpopulações diferentes. Um mecanismo para este fim já foi proposto em [Ahn et al., 2004], onde o operador foi chamado de cruzamento uniforme de PVs. Este operador funciona de forma análoga ao cruzamento uniforme de indivíduos no sGA. Dois vetores são combinados e um vetor temporário é gerado selecionando-se, aleatoriamente, para cada gene, de que vetor "pai" vai ser obtida cada proporção binomial. Um novo indivíduo é então gerado a partir deste vetor temporário.

Experimentos mostrados no capítulo 6 ilustram que o esquema proposto é capaz de agrupar indivíduos de acordo com o bloco construtor que possuem já nas primeiras etapas do processo evolutivo. Por meio da seleção, apenas indivíduos que possuam os blocos construtores iniciais vão permanecendo na população. Cada grupo se especializa em um bloco construtor específico. A combinação de blocos construtores, entretanto, não é realizada adotando esta abordagem. O operador de cruzamento uniforme de PVs [Ahn et al., 2004] não é capaz de combinar as informações contidas nos vetores adequadamente, não preservando a estrutura do problema.

Assim, confirma-se a necessidade da adoção de algum mecanismo adicional, conforme antecipado por [Pelikan e Goldberg, 2000]. Este mecanismo procura adquirir informações sobre a quantidade de informação que cada grupo carrega.

5.3 Combinação blocos construtores e aprendendo sobre a estrutura do problema

Esta seção descreve um mecanismo capaz de adquirir informações sobre a estrutura do problema a partir dos modelos binomiais aprendidos, ao longo do processo evolutivo, por um EDA de primeira ordem com agrupamentos. Estas medidas permitem

decidir cuidadosamente a melhor forma de combinar vetores de probabilidade (PVs) relacionados a subpopulações.

Dado que o modelo de cada grupo contém informações relevantes sobre uma parte da estrutura do problema, então a combinação de PVs deve, de alguma forma, combinar blocos construtores. Esta combinação de partes importantes é o que guia EDAs e o o sGA durante a exploração do espaço de busca [Pelikan e Goldberg, 2000]. Por isso a combinação entre grupos não deveria ser evitada, mas sim realizada criteriosamente.

O operador proposto é fundamentado na teoria da informação. Da mesma forma que o cruzamento uniforme de PV, o novo operador também opera sobre dois PVs "pai", que podem ter sido selecionados aleatoriamente entre os vetores existentes. Um vetor combinado, temporário, é gerado, a partir do qual um novo indivíduo é amostrado.

A diferença está no critério adotado para selecionar de qual "vetor-pai" cada proporção binomial será obtida. Uma medida da teoria da informação guia esta escolha de forma muito intuitiva. Já que se deseja manter a porção mais informativa de cada PV pai, então basta comparar qual dos pais é mais informativo para cada um dos genes. Com isso se obtém algo semelhante a um "conceito" associado ao grupo, o que de certa forma define aquele grupo quando comparado a um outro. Assim, o operador é chamado de combinação guiada por conceito, ou simplesmente cg-combinação.

Durante o cruzamento, dois vetores pais – A e B – são selecionados aleatoriamente, e proporcionalmente ao *fitness* médio dos indivíduos presentes nos grupos correspondentes. Um PV temporário, a partir do qual o novo indivíduo será gerado, é obtido tomando proporções de cada posição j a partir de um dos pais $\hat{\pi}_{A,j}$ ou $\hat{\pi}_{B,j}$. Uma medida da teoria de informação é usada de modo a guiar a escolha sempre no sentido do pai mais informativo para cada gene. Seja

$$\hat{h}_{\cdot,j} = -\sum_{q \in \{0,1\}} \hat{p}_{j,q} log_2(\hat{p}_{j,q}) \tag{5.1}$$

a entropia da distribuição do gene j, sendo que $\hat{p}_{j,q} = P(x_j = q)$ é a proporção de

indivíduos que possuem o valor q para o gene j em toda a população ($q \in \{0, 1\}$). Da mesma forma, seja

$$\hat{h}'_{i,j} = -\sum_{q \in \{0,1\}} \hat{p}'_{i,j,q} log_2(\hat{p}'_{i,j,q}) \tag{5.2}$$

a entropia da distribuição do mesmo gene j sem levar em conta o grupo i quando da estimação das proporções. Estas proporções são calculadas como $\hat{p}'_{i,j,q} = P(x_j = q \mid c(\boldsymbol{x}) \in \{1, 2, \ldots, k\} - \{i\})$, sendo que $c(\boldsymbol{x})$ representa o grupo ao qual pertence o indivíduo $\boldsymbol{x}$.

A medida de quão informativo é um grupo i para um gene j é, portanto, a diferença entre a entropia da distribuição do gene j antes e depois de observar o grupo i:

$$\hat{w}_{i,j} = \hat{h}_{i,j} - \hat{h}'_{i,j} \tag{5.3}$$

Assim, a decisão sobre como construir o PV temporário torna-se simples: basta escolher para cada gene j o pai que tiver o $\hat{w}_{i,j}$ maior. Cada posição v_j do PV temporário v é definida como

$$v_j = \begin{cases} \hat{\pi}_{A,j}, & se\ \hat{w}_{A,j} > \hat{w}_{B,j} \\ \hat{\pi}_{B,j}, & caso\ contrário \end{cases} \tag{5.4}$$

Depois do cálculo de v um novo indivíduo é gerado amostrando independentemente a partir de cada posição de v. A figura 5.3 mostra um cenário com $k = 3$ grupos e com 12 variáveis. A figura 5.4 ilustra a criação de v durante um cruzamento, para um pequeno problema com 4 variáveis.

O operador da cg-combinação confia na capacidade do algoritmo de agrupamento de manter indivíduos que possuem a mesma subestrutura juntos no grupo e procura extrair a informação que foi adquirida por meio deste agrupamento a partir das matrizes $\hat{\Pi} = (\hat{\pi}_{i,j})$ e $\hat{W} = (\hat{w}_{i,j})$. Espera-se que um grupo seja informativo para todos os genes que definem o seu bloco construtor associado se a maior parte dos indivíduos naquele grupo possuírem aquele bloco.

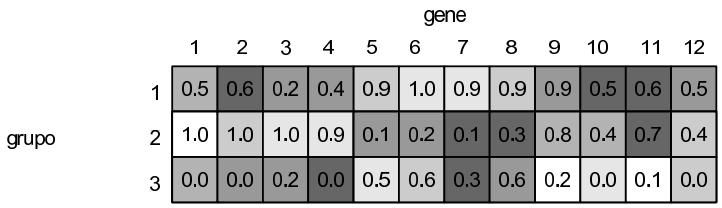


Figura 5.3: Um possível conjunto de proporções binomiais $\hat{\pi}_{i,j}$s e seus respectivos $\hat{w}_{i,j}$s em um certo momento do processo evolutivo, para um problema artificial. Os valores inscritos são os $\hat{\pi}_{i,j}$s, enquanto que os tons de cinza representam os $\hat{w}_{i,j}$ em uma escala que vai do preto (mínimo) até o branco (máximo).

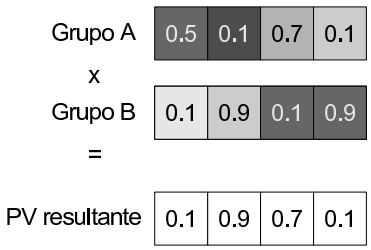


Figura 5.4: Geração de um PV durante uma operação de cruzamento entre grupos. $\hat{\pi}_{A,.}$, $\hat{\pi}_{B,.}$ e seus respectivos $\hat{w}_{i,j}$s são mostrados usando a mesma convenção de cores que na figura 5.3 (exceto o PV resultante que não possui $\hat{w}_{i,j}$s).

Se a combinação de blocos resultar em outros blocos ainda melhores, então se espera que os novos indivíduos que possuam estes blocos combinados passem a ser também agrupados, de modo que novos grupos emerjam como resultado deste processo, e o algoritmo prossegue dessa forma obtendo subestruturas cada vez maiores.

Cada grupo define um modelo probabilístico e os genes são assumidos como condicionalmente independentes, dado o rótulo do grupo. Assim, apenas dependências entre o gene e o rótulo do grupo são consideradas, de modo que o modelo resultante está limitado a estatísticas de segunda ordem apenas. Estas estatísticas são obtidas nos $\hat{h}_{i,j}$s e $\hat{h}'_{i,j}$s.

O aprendizado de ligação é realizado pelo aprendizado de agrupamentos, desde o menor bloco construtor até o maior. Inicialmente, cada grupo aprende um esquema. Medir a quantidade de informação de cada grupo para cada variável é útil apenas para preservar o bloco construtor, mas a informação já foi adquirida pelo aprendizado de agrupamentos. A quantidade de informação trazida por um grupo para a distribuição de uma variável deve estar relacionada com a probabilidade de que aquela variável

faça parte do bloco construtor aprendido por aquele agrupamento.

5.4 Aprendizado incremental

Nenhum dos algoritmos evolutivos baseados em inferência (EDAs) aborda de forma explícita a característica dinâmica e incremental do processo. Nestes algoritmos um novo modelo é construído a cada geração, a partir dos indivíduos mais recentemente gerados, podendo, eventualmente, considerar também indivíduos mais antigos. Em uma perspectiva de aprendizado em ambientes dinâmicos, com deslocamento de conceito [Gama et al., 2005], se pode afirmar que o tipo de aprendizado realizado pelos algoritmos baseados em inferência atualmente disponíveis na literatura é caracterizado como aprendizado em lote, ou não-incremental, já que novos modelos são aprendidos para sucessivas populações (que são entendidas como sucessivas janelas). Embora esta possa ser considerada uma abordagem válida para aprendizado em ambientes dinâmicos, existem outras formas mais eficientes de abordar este problema.

Certamente o paradigma de aprendizado incremental é a base das propostas mais eficientes, construídas recentemente para realizar aprendizado em ambientes dinâmicos [Gama et al., 2005]. Desta forma, a presente proposta tem como objetivo propor a adoção do aprendizado incremental, mais adequada a ambientes dinâmicos, considerando que o processo evolutivo possui muitas características deste tipo de ambiente.

A confiança em técnicas incrementais baseia-se na natureza dinâmica do processo de computação evolutiva. Nesta proposta, o modelo deve ser apenas corrigido após a apresentação de cada instância. O que se espera é verificar uma possível redução de custo computacional, assumindo que ocorre um maior aproveitamento da informação adquirida ao longo da execução do algoritmo.

O algoritmo proposto aqui, chamado de φ-PBIL (aprendizado incremental baseado em população e guiado por conceitos), segue uma abordagem incremental na qual um único indivíduo é gerado a cada iteração, sem a adoção das sucessivas "gerações" como é muito freqüente em computação evolutiva. O algoritmo é um EDA de baixa

ordem que segue uma abordagem incremental no agrupamento de indivíduos e adota o operador cg-combinação para combinar PVs. O pseudocódigo do φ-PBIL está no algoritmo 2.

Algoritmo 2 φ-PBIL

// Inicialização:
Gera uma população inicial aleatória de tamanho N_0.
Calcula o *fitness* dos indivíduos
Seleciona os $N_w (N_w < N_0)$ melhores.
// Aprendizado:
Aprende grupos a partir da população.
Calcula vetores de probabilidade (PV) de proporções binomiais para cada grupo.
Guarda os PVs na matriz $\hat{\Pi} = (\hat{\pi}_{i,j})$.
Calcula medidas de informação $\hat{W} = (\hat{w}_{i,j})$.
enquanto critérios de convergência não forem satisfeitos **faça**
  // Gera um novo indivíduo H:
  Cria H escolhendo aleatoriamente entre um dos seguintes procedimentos:
   (i) amostra diretamente de um dos PVs
    ou (ii) aplica cg-combinação, utilizando as matrizes $\hat{\Pi}$
      e $\hat{W}$, e amostra do PV combinado.
  // Seleção:
  Calcula o *fitness* do novo indivíduo H.
  se H não for pior que o pior indivíduo na população **então**
    Apague este pior indivíduo.
    Insira H na população.
  fim se
  // Aprendizado:
  Atualize grupos e matrizes $\hat{\Pi}$ e $\hat{W}$.
fim enquanto

O algoritmo inicia a partir de uma população criada aleatoriamente, de tamanho N_0. Desta, apenas os N_w melhores são selecionados e o algoritmo prossegue operando sempre sobre N_w indivíduos.

Uma quantidade fixa de k grupos são mantidos e atualizados continuamente. Sempre que um novo indivíduo é gerado ele passa por um critério de seleção incremental. Se ele for melhor que o pior indivíduo atualmente na população então a população é atualizada, removendo o pior indivíduo e inserindo o novo. Neste caso, as hipóteses de agrupamento atuais devem ser atualizadas. Para isso, uma atualização incremental do algoritmo *k-means* é executada, a partir dos centróides atuais. Assim, os grupos não

precisam ser reaprendidos completamente a cada nova inserção de um indivíduo.

Neste processo incremental pode acontecer de que algum dos grupos fique vazio. Isto é resultado do processo de seleção: um dos grupos, que estava associado a um esquema sub-ótimo, passa a não conter mais indivíduos e seu centróide não atrai novos indivíduos aceitos pelo critério de seleção incremental. Neste caso, o *k-means* é reinicializado.

Cada grupo define uma subpopulação e, como apenas variáveis binárias são admitidas, então os modelos binomiais são obtidos a partir dos próprios centróides resultantes do *k-means* . Cada proporção $\hat{\pi}_{i,j}$ denota a fração dos indivíduos que possuem o valor 1 em cada gene j para cada grupo i.

De modo similar a outros EDAs, novos indivíduos são gerados amostrando-se do modelo adotado. No caso do φ-PBIL, existem duas possibilidades: ou amostrar diretamente de um PV, ou obter um PV temporário combinando dois modelos. Esta decisão é tomada aleatoriamente cada vez que um novo indivíduo é gerado, dado um parâmetro p_c que determina a probabilidade de se efetuar cruzamentos. Quando se está amostrando de um único PV, este é escolhido aleatoriamente e proporcionalmente ao *fitness* médio dos indivíduos pertencentes ao grupo.

Caso a decisão seja por combinar PVs, o operador *default* é a cg-combinação, que foi apresentada na seção anterior. Dois PVs são selecionados também aleatoriamente e um PV temporário é gerado a partir da combinação destes "pais". Alguns experimentos realizados no capítulo 6 substituem a cg-combinação por outros operadores, ilustrando a importância da cg-combinação para o funcionamento do φ-PBIL.

O critério de término do algoritmo é baseado na saturação dos PVs. O algoritmo termina quando todos os $\hat{\pi}_{i,j}$s atingem valores acima de $0,95$ ou abaixo de $0,05$.

5.5 Adição de memória

Em problemas hierárquicos existe interação entre os blocos construtores [Watson et al., 1998]. A combinação de blocos construtores primordiais leva a blocos com uma

contribuição para o *fitness* maior que a contribuição dos blocos originais. Isto pode fazer com que, rapidamente, deixem de existir, na população, indivíduos com apenas um bloco, já que indivíduos com mais de um bloco combinado devem dominar rapidamente. Este comportamento acontece no algoritmo proposto, como em outros algoritmos de computação evolutiva. A figura 6.2 ilustra este aspecto. Sabe-se que esta característica do processo evolutivo, em geral, é uma conseqüência da já bem conhecida disputa entre seleção e inovação [Goldberg, 1998], que são claramente forças contrárias, porém complementares.

O efeito destas forças na dinâmica do φ-PBIL é o da rápida convergência para esquemas com mais de um bloco primordial. Assim, a informação sobre os blocos primordiais, ou mesmo sobre as combinações menores destes blocos, rapidamente se perde sem que se tenha tido tempo suficiente para explorar um número suficiente de combinações destes blocos primordiais. O mesmo acontece em qualquer nível de um problema hierárquico.

A sobreposição de blocos construtores, embora não seja problemática para o φ-PBIL, também merece algum cuidado. Se dois grupos são informativos para o mesmo gene, uma das estruturas que se sobrepõe deve ser preservada a cada cruzamento. Então é importante que o algoritmo mantenha as informações sobre blocos menores para que se possa explorar ao máximo as combinações destes.

Seria desejável, portanto, que o algoritmo mantivesse informações sobre níveis anteriores da estrutura enquanto explora combinações superiores. Foi adotada uma solução baseada em incorporar memória sobre uma hipótese anterior de agrupamento, a qual constitui toda a estatística necessária para gerar novos indivíduos. No momento de gerar um indivíduo, seleciona-se de qual modelo ele será gerado: do antigo ou do novo.

O mecanismo funciona da seguinte forma: as matrizes $\hat{W} = (\hat{w}_{i,j})$ e $\hat{\Pi} = (\hat{\pi}_{i,j})$ correspondentes a uma hipótese antiga de agrupamento são mantidas, na forma das matrizes $\{\hat{\Pi}_{old}$ e $\hat{W}_{old}\}$. Sempre que um novo indivíduo vai ser gerado por combinação

de PVs é selecionado aleatoriamente entre as hipóteses antiga e nova. O parâmetro p_{old} determina esta probabilidade de se selecionar a hipótese antiga.

O número de indivíduos gerados e aceitos, de cada hipótese, representa o desempenho da hipótese. Existem dois registros de desempenho, um para cada hipótese: g_{old} e g_{new}. Se o indivíduo gerado é selecionado, o registro de desempenho correspondente à hipótese a partir da qual ele foi gerado é incrementado. Estes registros servem para controlar a atualização da hipótese antigo. Sempre que g_{new} superar g_{old}, então $\hat{W}_{old} \leftarrow \hat{W}$, $\hat{\Pi}_{old} \leftarrow \hat{\Pi}$, $g_{old} \leftarrow g_{new}$ e $g_{new} \leftarrow 0$. Ou seja, a hipótese atual passa a ser a "nova" hipótese antiga.

5.6 Incerteza em proporções binomiais

Na maioria dos EDAs não existe o mecanismo tradicional de mutação. A busca local é realizada apesar disso, pois a amostragem feita a partir dos modelos inferidos gera indivíduos próximos, mas não necessariamente idênticos aos melhores indivíduos da população.

No caso de variáveis binárias, por exemplo, mesmo que a grande maioria dos indivíduos possuam o valor 1 em um gene, ainda resta alguma chance de se gerar um 0 naquele mesmo gene, desde que a proporção estimada não seja igual a 100%.

O estimador de máxima verissimilhança (MLE) da proporção binomial é simplesmente a fração de sucessos. No caso de EDAs esta fração deve ser interpretada como a proporção de 1s em um gene, ou

$$\hat{\pi}_{i,j} = \frac{\sum_{c(x)=i} x_j}{n_i} \tag{5.5}$$

sendo que $\sum_{c(x)=i} x_j$ é o número de indivíduos no grupo i que possuem o valor 1 no gene j e n_i é o número de indivíduos no grupo i.

Este estimador satura em 0% ou em 100%, no caso de todos os indivídus no grupo possuírem um mesmo valor (todos 0 ou todos 1) em um certo gene. Isto elimina a

chance de se gerarem novos indivíduos possuindo o valor complementar ao valor saturado para aquele gene, a partir daquele grupo. Naturalmente, esta propriedade é indesejável pois limita a busca local.

Um forma de incorporar incerteza na estimação de proporções binomiais é usar o estimador de Wilson, que foi revisado em [Agresti e Coull, 1998]. Usando este estimador, a proporção binomial é estimada por meio da expressão

$$\hat{\pi}_{i,j} = \frac{\sum_{c(x)=i} x_j + 2}{n_i + 4} \tag{5.6}$$

que deve ser usada ao invés do estimador de máxima verossimilhança (5.5).

Como um exemplo, se todos os 100 indivíduos em um grupo possuírem o valor 1 em um gene, então usando a expressão 5.6 a proporção binomial estimada seria de 98%, que permitiria ainda 2% de chance de ser gerado um valor 0 naquele gene.

O uso do estimador de Wilson cada vez que um novo indivíduo é gerado é controlado pelo parâmetro p_w. O efeito esperado deste estimador é semelhante o mesmo da mutação em busca local. Outra forma possível de se incorporar incerteza seria por meio da inferência Bayesiana. Neste paradigma, o parâmetro é entendido como sendo uma variável aleatória e uma distribuição de probabilidade para o parâmetro é especificada *a priori*, que representa o grau de incerteza que se tem a respeito da distribuição do parâmetro. A distribuição *a posteriori* combina a informação da *priori* com a evidência fornecida pelos dados.

5.7 Parâmetros do algoritmo

Todos os parâmetros do algoritmo φ-PBIL constam na tabela 5.1. Esta seção apresenta um estudo empírico que permite sugerir valores *a priori* para alguns dos parâmetros. Este conjunto de valores, também chamados de *default*, são adotados como padrão para a execução do algoritmo em um problema geral, sendo possível alterar qualquer dos parâmetros para, eventualmente, obter um comportamento mais bem ajustado para

Tabela 5.1: Parâmetros do φ-PBIL

Parâmetro	Descrição	Valor *default*
N_0	tamanho da população inicial	–
N_w	tamanho da população de trabalho	–
k	número de grupos	–
p_c	probabilidade de cruzamento entre grupos	50%
p_{old}	probabilidade de adotar a hipótese antiga de agrupamento	50%
p_w	probabilidade de usar o estimador de Wilson	50%

alguma classe específica de problemas. Para alguns parâmetros não há valor *default*: tamanho da população inicial N_0, tamanho da população de trabalho N_w e número de grupos k. Estes três parâmetros do algoritmo estão relacionados ao tamanho do problema e, por isso, não é adequado atribuir algum valor *default*.

Valores *default* foram encontrados para os demais parâmetros após investigação empírica, a qual é descrita a seguir. Os parâmetros p_c (probabilidade de cruzamento), p_{old} (probabilidade de adotar a hipótese antiga de cruzamento ao invés da atual) e p_w (probabilidade de usar o estimador de Wilson ao invés do MLE) determinam o comportamento do algoritmo.

A investigação sobre os valores *default* destes parâmetros é feita usando três problemas bastante representativos: HIFF [Watson et al., 1998], função armadilha concatenada de ordem 5 [Pelikan et al., 1999] e particionamento de grafos [Peña et al., 2005]. Parte-se já da hipótese de que p_c, p_{old} e p_w devem ser todos 50%. As instâncias adotadas são Pshuff64, Ptrapfive50 and Pcatring42, todas relativamente pequenas mas sufientes para revelar diferenças de comportamento do algoritmo sob diferentes configurações de parâmetros. Pcatring42 e Pshuff64 são globalmente multimodais, possuindo 6 e 2 ótimos globais respectivamente enquanto Ptrapfive50 tem um único ótimo global.

Os três parâmetros relacionados ao tamanho do problema são mantidos fixos para todas as rodadas de cada problema neste experimento: $N_0 = 2500$, $N_w = 250$ e $k = 10$ para Pcatring42 e $N_0 = 3000$, $N_w = 300$ e $k = 15$ para Pshuff64 e Ptrapfive50. Verificou-

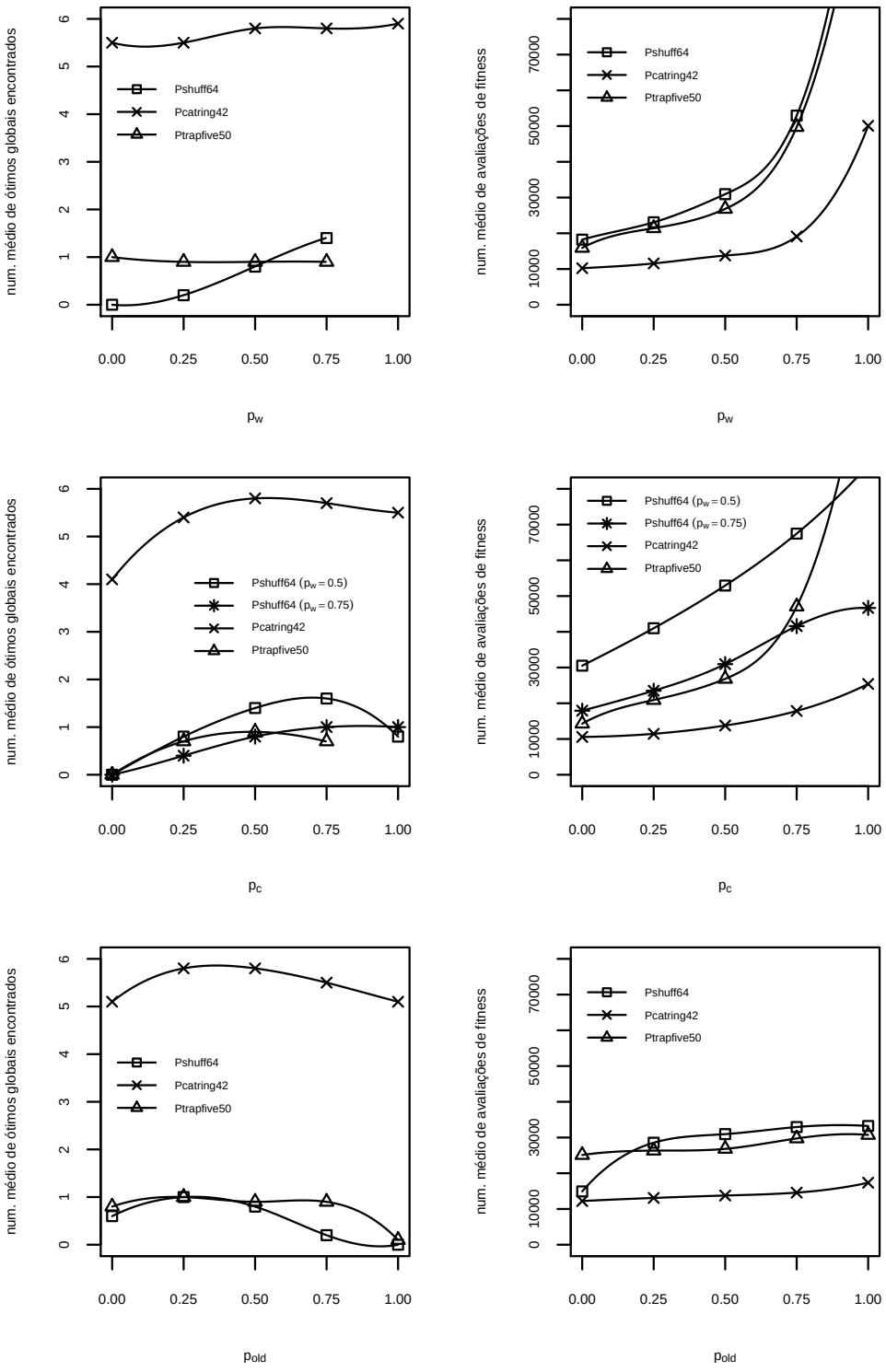


Figura 5.5: Avaliação empírica da sensibilidade do φ-PBIL a três parâmetros p_c, p_{old} e p_w, relatando o número médio de ótimos globais encontrados e mantidos e o número médio de avaliações de *fitness* realizadas até a convergência.

se previamente que estes valores estão próximos aos mínimos necessários ao algoritmo para que seja capaz de encontrar todos os ótimos globais das instâncias consideradas aqui.

A figura 5.5 mostra os resultados desta investigação. Cada gráfico mostra a variação de um dos parâmetros considerados, de modo que os outros dois permanecem fixos. Caso não esteja explicitamente expresso em contrário, o valor dos outros dois parâmetros é 50%. Cinco valores (0%, 25%, 50%, 75% e 100%) para cada parâmetro são considerados em cada linha. Cada ponto no gráfico representa a média de 10 rodadas independentes do algoritmo (replicações). As curvas são resultado de interpolação por *splines*. Quando o número médio de avaliações de *fitness* excede 100.000 os respectivos segmentos de linha correspondentes em ambos os gráficos são truncados.

A mudança de cada um dos parâmetros pode afetar dramaticamente o comportamento do algoritmo para alguns dos problemas. Setando p_{old} para 0% ou 100% faz com que o desempenho no problema hierárquico Pshuff64 caia já que as hipóteses antigas e novas de agrupamento se complementam na combinação de informações sobre níveis subseqüentes da estrutura do problema. O desempenho em outros problemas é menos afetado por p_{old}, exceto com p_{old} em 100%, que causa a redução de desempenho em Ptrapfive50. Usar apenas hipóteses antigas parece ser prejudicial para o algoritmo, como esperado.

Ajustar a probabilidade de cruzamento – p_c – para zero afeta o desempenho do algoritmo para uma gama mais ampla de problemas já que este mecanismo é o mais responsável pela exploração. Fazendo com que p_c aumente muito também afeta negativamente a velocidade convergência para todas as instâncias testadas.

O parâmetro p_w também é negativamente relacionado à velocidade de convergência, mas valores menores para p_w também devem ser evitados, pelo menos para o HIFF, já que nenhum a solução global é encontrada para Pshuff64 com $p_w = 0$. A eficácia em outros problemas não foi afetada por este parâmetro.

Analisando os resultados e reconhecendo, a partir deles, que valores extremos para

todos os parâmetros devem ser evitados então o conjunto de valores *default* é escolhido como p_c, p_{old} e p_w todos em 50%. Outras possíveis combinações que igualmente evitam extremos são aquelas onde p_c, p_{old} e p_w pertencem ao intervalo $[25\%, 75\%]$.

CAPÍTULO 6

AVALIAÇÃO EMPÍRICA

Neste capítulo são apresentados os resultados da avaliação empírica do algoritmo proposto. Este estudo fornece subsídios para avaliar a validade da hípótese investigada nesta tese. Mais especificamente, se pretende verificar a eficácia do mecanismo de aprendizado de ligação apresentado.

Os problemas adotados nesta avaliação são representativos das classes de problemas considerados importantes para avaliação de EDAs de alta ordem, por apresentarem estruturas de interação entre variáveis, de diversas formas. Os problemas incluem características enganadoras [Pelikan et al., 1999], simetria [Pelikan e Goldberg, 2000], hierarquia [Watson et al., 1998], multimodalidade global [Peña et al., 2005] e a presença de subestruturas sobrepostas [Yu et al., 2005]. É importante notar que não há na literatura relato de algum EDA de baixa ordem que seja capaz de resolver algum destes problemas. Todos os EDAs competentes, até o momento, adotam estatísticas de ordem superior a dois.

Quando o mecanismo de cg-combinação é substituído por um operador mais simples, como o cruzamento uniforme de PVs, o algoritmo não é capaz de encontrar o ótimo global em nenhum dos problemas testados. O operador de cg-combinação é fundamental para o funcionamento do algoritmo proposto.

Inicialmente, os dois mecanismos de combinação são comparados. A seguir, uma avaliação da escalabilidade do algoritmo é mostrada. Por fim, uma comparação entre φ-PBIL e UEBNA [Peña et al., 2005], para problemas globalmente multimodais e simétricos é detalhada.

6.1 Comparação de operadores de cruzamento

O primeiro experimento ilustra o comportamento do operador de cg-combinação, comparado ao cruzamento uniforme de PVs [Ahn et al., 2004]. O cruzamento uniforme mistura dois PVs aleatoriamente enquanto a cg-combinação escolhe cuidadosamente de que pai deve ser copiada cada proporção binomial, conforme descrito no capítulo 5. Neste experimento o algoritmo φ-PBIL foi rodado com os parâmetros *default*. Além disso, $k = 4$ grupos e os tamanhos de população foram $N_0 = N_w = 100$.

Antes de uma comparação mais quantitativa, o comportamento típico de ambos os operadores é ilustrado. Como o experimento é apenas ilustrativo então uma única rodada para cada operador é mostrada, usando o problema armadilha concatenada de ordem 5 com tamanho 15. Este é um problema separável e para cada subproblema existe um bloco construtor ótimo e um bloco sub-ótimo, enganoso. Os três blocos ótimos – denotados por BB0, BB1 e BB2 – devem ser encontrados e, subseqüentemente, combinados entre si para que se encontre a solução ótima.

Encontrar os ótimos por meio da detecção de similaridades é bastante simples para o algoritmo mesmo quando o operador de cruzamento uniforme é usado, até porque este operador não interfere no processo de identificação dos blocos primordiais. O difícil é combiná-los adequadamente. Esta rodada está ilustrada na figura 6.1. O número de avaliações de *fitness* é reportado como dimensão tempo na figura, já que o processo é incremental e não existem sucessivas "gerações" explicitamente. O eixo vertical mostra a proporção de indivíduos na população que possuem cada um dos blocos construtores.

Usando o cruzamento uniforme de PVs o algoritmo não é capaz de combinar blocos construtores eficientemente. O ótimo global, que é a combinação de BB0, BB1 e BB2, não é encontrado. Cada grupo se especializa, inicialmente, em um padrão específico de similaridade. Conforme esperado, cada grupo aprende um esquema. O grupo 0, por exemplo, se especializa em atrair indivíduos que possuem o bloco BB2. À medida que o processo evolui no tempo, a proporção de indivíduos gerados que possuem BB2

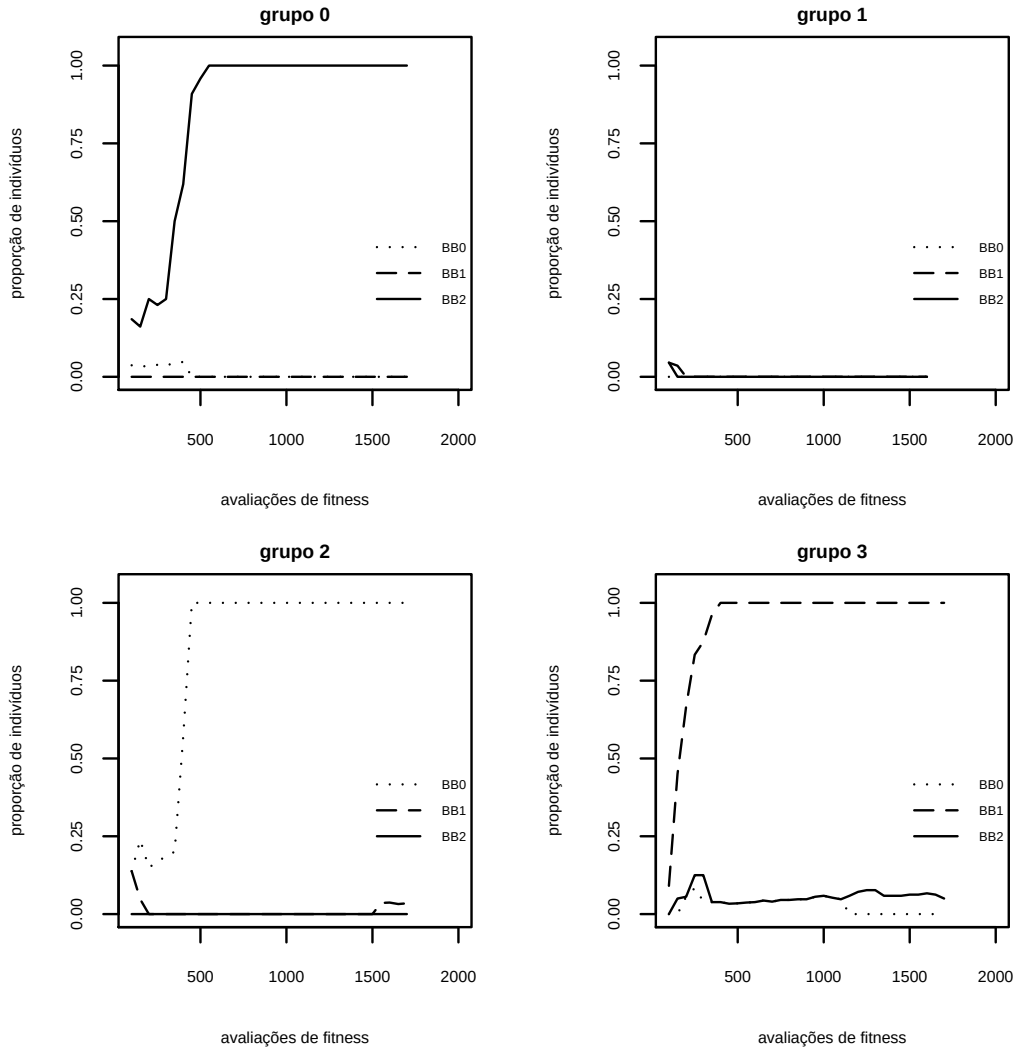


Figura 6.1: Uma rodada do φ-PBIL usando o cruzamento uniforme de PVs para um problema separável com três 3 blocos construtores (BB0, BB1 e BB2).

aumenta e estes indivíduos vão sendo atraídos para o próprio grupo 0, o que é natural.

Por volta da avaliação de *fitness* número 500, 100% dos indivíduos no grupo 0 possuem o bloco BB2. Uma dinâmica semelhante ocorre nos demais grupos – o grupo 2 se especializa em BB0 e o grupo 3 em BB1. Naturalmente que outras atribuições de grupos para BBs devem acontecer em outras rodadas, mas o comportamento geral se preserva. A dificuldade em encontrar a solução global, entretanto, se observa em outras rodadas.

Rodando o algoritmo φ-PBIL com o operador de cg-combinação para a mesma semente aleatória, conseqüentemente a mesma população inicial, leva a um comporta-

mento muito melhor, como ilustrado na figura 6.2. Nas primeiras etapas do processo (cerca de 400 avaliações de *fitness*) o mesmo tipo de especialização observado anteriormente é também verificado.

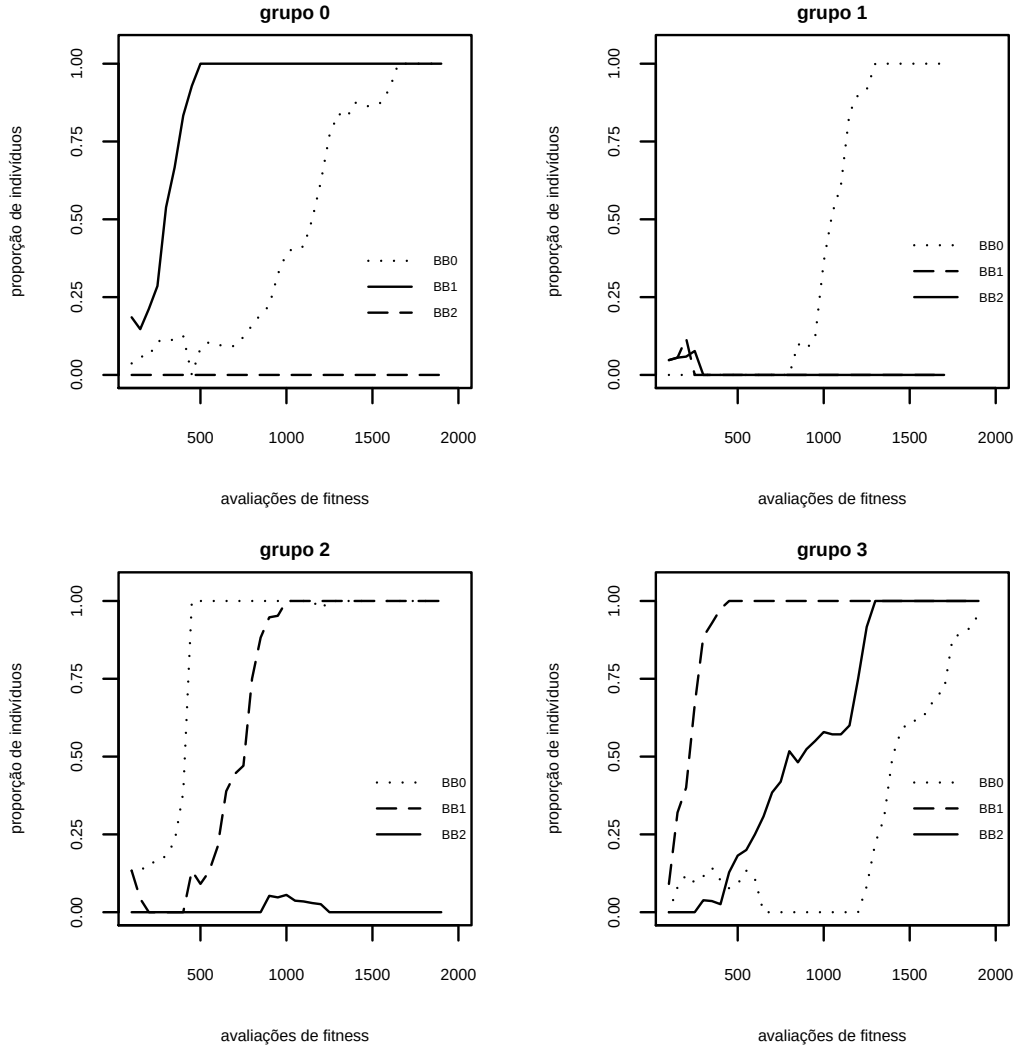


Figura 6.2: Uma rodada do φ-PBIL usando a cg-combinação para um problema separável com três 3 blocos construtores (BB0, BB1 e BB2).

A partir daí se observam diferenças. À medida que o número de combinações com sucesso aumenta, os grupos originais passam a atrair, além de indivíduos mais simples com um único bloco construtor, também indivíduos com dois ou três blocos construtores simultaneamente, resultantes destas combinações. Após cerca de 2000 avaliações de *fitness* o grupo 3 se especializa em indivíduos com 3 BBs, que corresponde ao ótimo global. Naquele momento as posições de todos os PVs já estão saturadas acima de $0,95$

ou abaixo de $0,05$, o que é a condição de convergência do algoritmo.

A capacidade de manutenção de diversidade é bastante eficiente pois, mesmo na convergência, ainda existem muitos indivíduos na população com combinações subótimas de blocos construtores.

Uma comparação mais quantitativa entre os dois operadores de combinação é apresentada na tabela 6.1.

Tabela 6.1: Comparação da cg-combinação com o cruzamento uniforme de PVs. Para cada problema o mesmo conjunto de parâmetros é usado para ambos operadores.

problema (tamanho)	φ-PBIL usando combinação guiada por conceitos	φ-PBIL usando combinação uniforme de PV
	% sucessos	% sucessos
HIFF misturado (128)	97%	0%
Armadilha-5 concatenada (100)	100%	0%
Armadilha-5 c. sobreposta (60)	100%	0%
Twomax (100)	100%	100%
problema (tamanho)	φ-PBIL usando combinação guiada por conceitos	φ-PBIL usando combinação uniforme de PV
	Aval.$\pm$ dp	Aval.$\pm$ dp
HIFF misturado (128)	105134$\pm$ 11612	81646$\pm$ 13474
Armadilha-5 concatenada (100)	90474$\pm$ 7203	49391$\pm$ 7366
Armadilha-5 c. sobreposta (60)	55649$\pm$ 4207	23729$\pm$ 1048
Twomax (100)	4825$\pm$ 216	4867$\pm$ 318

Os mesmos parâmetros foram usados, para cada operador, em ambos conjuntos de rodadas. Um total de 30 rodadas independentes do algoritmo foram executadas para cada problema. A fração de sucessos (% sucessos), onde uma rodada com sucesso é considerada como aquela onde pelo menos um ótimo global é encontrado, e o número médio de avaliações de *fitness* até a convergência (Aval.) e seu respectivo desvio padrão (dp) são relatados.

Quatro problemas representativos foram escolhidos: HIFF misturado, armadilha-5 concatenada, armadilha-5 concatenada sobreposta e Twomax, com instâncias de tamanho 128, 100, 60 e 100 respectivamente.

Quando é usada a cg-combinação, pelo menos um ótimo global é encontrado em

97% das rodadas para o HIFF e em 100% das rodadas para os demais problemas. De fato, quando este operador é usado todos os ótimos globais do HIFF são encontrados em 93% das rodadas, e em 100% das rodadas para os outros problemas. O cruzamento uniforme de PVs não foi capaz de encontrar o ótimo global em nenhum dos problemas testados, exceto no caso do Twomax que é um problema bastante simples e foi beneficiado pelo agrupamento realizado para que seus dois ótimos globais pudessem ser encontrados.

Nenhum dos resultados para o cruzamento uniforme são surpreendentes, pois já se sabe que operadores de cruzamento que não respeitam a estrutura do problema estão condenados a falhar em problemas estruturados como estes adotados aqui. Quando se desliga o cruzamento por completo os resultados obtidos são semelhantes, de modo que realizar o cruzamento uniforme não proporciona ganho em termos de exploração de combinações de subestruturas.

6.2 Escalabilidade em um problema enganoso

A escalabilidade de um algoritmo diz respeito a sua complexidade computacional. Se espera que um algoritmo de otimização tenha escalabilidade subquadrática para que possa ser aplicado a problemas de interesse prático, com instância de grande dimensão.

Este experimento testa a escalabilidade do φ-PBIL no problema armadilha-5 concatenada. São executadas 30 rodadas independentes para cada tamanho do problema. Os tamanhos testados foram $\lambda = 100$, 125, 150, 175 e 200. A população de trabalho N_w foi ajustada como sendo proporcional ao tamanho do problema como $N_w = 5\lambda$, para se verificar a hipótese de escalabilidade linear. A população inicial é $N_0 = 15N_w$. O número de grupos k também foi tomado como proporcional ao tamanho do problema: $k = \lambda/5$.

A figura 6.3 mostra os resultados deste experimento. Na mesma figura, um estudo semelhante realizado para o algoritmo BOA, conduzido em [Pelikan, 2005], também é mostrado. Os resultados mostram que ambos os algoritmos tem um ótimo comporta-

mento de escalabilidade neste problema. A hipótese de escalabilidade linear, embora precise ser melhor verificada inclusive analiticamente, não deve ser descartada. Modelos lineares resultam coeficientes de determinação altos: $R^2 = 0,9992$ para φ-PBIL e $0,9846$ para o BOA.

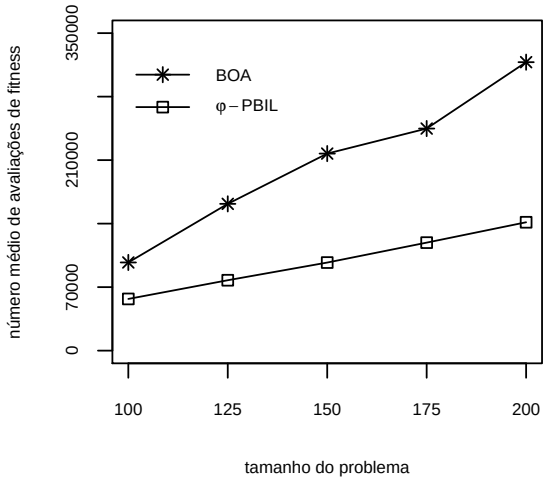


Figura 6.3: Escalabilidade de φ-PBIL e BOA para alguns tamanhos do problema armadilha concatenada de ordem 5. Dados para o BOA extraídos de [Pelikan, 2005]. O número médio de avaliações de *fitness* até a convergência é reportado no eixo vertical.

É importante mencionar que a diferença dos dois algoritmos, em termos de valores absolutos, não está sendo aqui avaliada. Embora o φ-PBIL tenha exigido um menor número de avaliações de *fitness* em todos os tamanhos de problema, quando comparado ao BOA, isto não significa que o BOA não possa apresentar um resultado melhor através de alterações no ajuste de parâmetros. O que se pretende avaliar é a complexidade computacional em função do tamanho do problema, a qual parece ser linear nos dois algoritmos, para este problema.

6.3 Problemas hierárquicos e sobreposição de subestruturas

O segundo experimento apresentado na seção 6.1 ilustra a capacidade do algoritmo proposto ao resolver problemas hierárquicos. Uma discussão sobre o comportamento

do algoritmo proposto nesta classe de problemas é relevante aqui.

O algoritmo φ-PBIL parece resolver o problema das ordens crescentes de interação em problemas hierárquicos, apresentado na seção 4.3. Uma vez que um grupo detecta adequadamente a existência de um bloco construtor A, os indivíduos que possuem este BB permanecerão na população enquanto possuírem *fitness* alto para os padrões da população. Se a combinação de A com outro bloco construtor B (que pode ou não se sobrepor a A) for benéfica, então novos indivíduos serão gerados respeitando A e B, originando um novo BB de ordem maior. Quando uma quantidade suficiente de indivíduos for gerada e aceita um grupo será formado, agrupando estes indivíduos. O grupo formado será, automaticamente, muito informativo para todas as variáveis que compõem o novo bloco construtor resultante de $A \cup B$, logo, cruzamentos deste com outros BBs devem preservar $A \cup B$ com alta probabilidade.

Da mesma forma, um problema com estruturas sobrepostas também foi avaliado no mesmo experimento. A sobreposição de blocos não inviabiliza o funcionamento do algoritmo. Dois blocos primordiais A e B, que se sobrepõem, vão estar associados a grupos distintos. Ocorre que os dois grupos serão igualmente informativos para aquelas variáveis que compõem a sobreposição, o que implica em um aumento na probabilidade de que o cruzamento entre estes dois grupos resulte em rompimento dos respectivos blocos construtores. Entretanto, o cruzamento entre grupos associados a blocos que não se sobrepõe preserva a informação de cada bloco, o que permite a sucessiva combinação dos blocos construtores neste caso.

6.4 Otimização multimodal e simetria

Inicialmente é apresentado, na figura 6.4, o resultado de um experimento que visa ilustrar o comportamento do algoritmo quando em um problema simétrico simples: o Twomax. Uma instância de tamanho 50 é considerada. Cada gráfico representa um momento do processo evolutivo, a partir de uma população inicial aleatória (não mostrada). O tempo t é contado em avaliações de *fitness*. O gráfico mostra as freqüências,

Tabela 6.2: Eficácia e eficiência do φ-PBIL e UEBNA para 10 instâncias do problema da bisseção de grafos.

Problema	EDA	Ótimo±sd	Aval.±sd
Pgrid16 (2 picos)	UEBNA $k = 4$	2.0±0.0	51400±2366
	φ-PBIL $k = 5$	2.0±0.0	10126±606
Pgrid36 (2 picos)	UEBNA $k = 2$	2.0±0.0	85600±8462
	φ-PBIL $k = 5$	2.0±0.0	28963±10754
Pgrid64 (2 picos)	UEBNA $k = 4$	2.0±0.0	124900±3479
	φ-PBIL $k = 10$	2.0±0.0	64245±10999
Pcat28 (2 picos)	UEBNA $k = 2$	2.0±0.0	57100±2846
	φ-PBIL $k = 5$	2.0±0.0	14311±1299
Pcat42 (2 picos)	UEBNA $k = 2$	2.0±0.0	73900±1449
	φ-PBIL $k = 10$	2.0±0.0	29714±3644
Pcat56 (2 picos)	UEBNA $k = 4$	2.0±0.0	96400±2366
	φ-PBIL $k = 10$	2.0±0.0	46151±4362
Pcatring28 (4 picos)	UEBNA $k = 2$	4.0±0.0	54700±949
	φ-PBIL $k = 5$	4.0±0.0	12694±853
Pcatring56 (4 picos)	UEBNA $k = 8$	3.8±0.4	96400±1897
	φ-PBIL $k = 10$	3.9±0.3	48837±12115
Pcatring42 (6 picos)	UEBNA $k = 6$	5.9±0.3	75700±3302
	φ-PBIL $k = 15$	6.0±0.0	32361±1513
Pcatring84 (6 picos)	UEBNA $k = 10$	4.8±0.8	121000±3162
	φ-PBIL $k = 20$	5.7±0.7	84539±9300

na população, de indivíduos com cada um dos possíveis valores para o total de 1s (de 0 a 50).

O agrupamento tem o efeito de separar a população em dois grupos (já que $k = 2$ neste caso). Com isso, em cada grupo ocorre a convergência para um ótimo global.

Um experimento mais quantitativo foi também conduzido. Este valida φ-PBIL em um problema de otimização combinatória multimodal relevante e complexo: o particionamento de grafos. Algumas das instâncias verificadas apresentam uma estrutura de interação entre variáveis. Particularmente, as topologias em anel (Pcatring56, Pcatring42 e Pcatring84) são desafiadoras pois a estrutura de interação entre as variáveis é clara. Em uma codificação de um gene binário por nó existem, para muitas das instâncias, grupos de 7 nós que devem ser identificados para que se possa explorar adequadamente a estrutura do problema. Estas características exigem tanto uma boa manutenção de diversidade quanto um esquema efetivo de aprendizado de ligação.

Uma comparação entre φ-PBIL e UEBNA é descrita. O algoritmo UEBNA mostrou-se competente na resolução desta classe de problemas, tendo recentemente [Peña et al., 2005] superado outros EDAs competentes resolvendo as mesmas instâncias de particionamento de grafos mostradas aqui. Experimentos reportados em [Peña et al., 2005] mostram que UEBNA atinge um desempenho muito superior à do EBNA, que é baseado na indução supervisionada de redes Bayesianas, e não incorpora rótulos de grupo, como faz o UEBNA. A codificação é adotada conforme a descrita no capítulo 2, incluindo o operador de reparo detalhado lá.

Todos os resultados para o UEBNA apresentados aqui foram extraídos de [Peña et al., 2005]. São realizadas 10 rodadas independentes de cada algoritmo para cada instância de problema. Um conjunto fixo de parâmetros foi fixado para esta avaliação: $p_w = 75\%$, $p_{old} = 25\%$ e $p_c = 50\%$. Quando os parâmetros *default* são usados, os resultados são um pouco inferiores. O tamanho da população inicial é fixado em $N_0 = 4.000$, o mesmo valor que o usado pelo UEBNA. Cabe aqui uma observação, já que a população de trabalho do φ-PBIL pode ser tomada como uma fração da população inicial. Nos experimentos com φ-PBIL, a população de trabalho é de $N_w = 500$ indivíduos para todos os problemas. Adotou-se como único parâmetro livre o k, que representa número de agrupamentos. É reportado sempre o valor de k que maximiza a eficiência na avaliação empírica, restrito a $k \in [2, 3, \ldots, 20]$. O mesmo procedimento foi tomado em relação ao UEBNA, que possui um parâmetro k análogo.

A tabela 6.2 resume os resultados. A média $\pm$ o desvio padrão do número de ótimos globais (picos) encontrados e mantidos são reportados como (*Ótimos*$\pm$*s.d.*). Da mesma forma, média e desvio padrão do número de avaliações de *fitness* até a convergência são reportados como (*Aval.*$\pm$*s.d.*). Todas as rodadas de ambos algoritmos encontraram pelo menos um ótimo global para todas as instâncias testadas.

Os resultados deixam claro que o φ-PBIL atinge convergência para todos os ótimos globais usando um número menor de avaliações de *fitness*. Para instâncias menores ambos algoritmos encontram todos os ótimos globais em todas as rodadas mas,

quando o tamanho das instâncias aumenta, o algoritmo φ-PBIL revela-se com melhor eficácia e eficiência. O número médio de ótimos encontrados é melhor que o do UEBNA, notadamente para Pcatring56, Pcatring42 e Pcatring84. Para Pcatring84, particularmente, a qual apresenta 6 ótimos globais, φ-PBIL encontra em média 5.7 ± 0.7 ótimos globais, enquanto UEBNA encontra 4.8 ± 0.8. Isto sugere que φ-PBIL apresenta boa escalabilidade também nesta classe de problemas combinatórios.

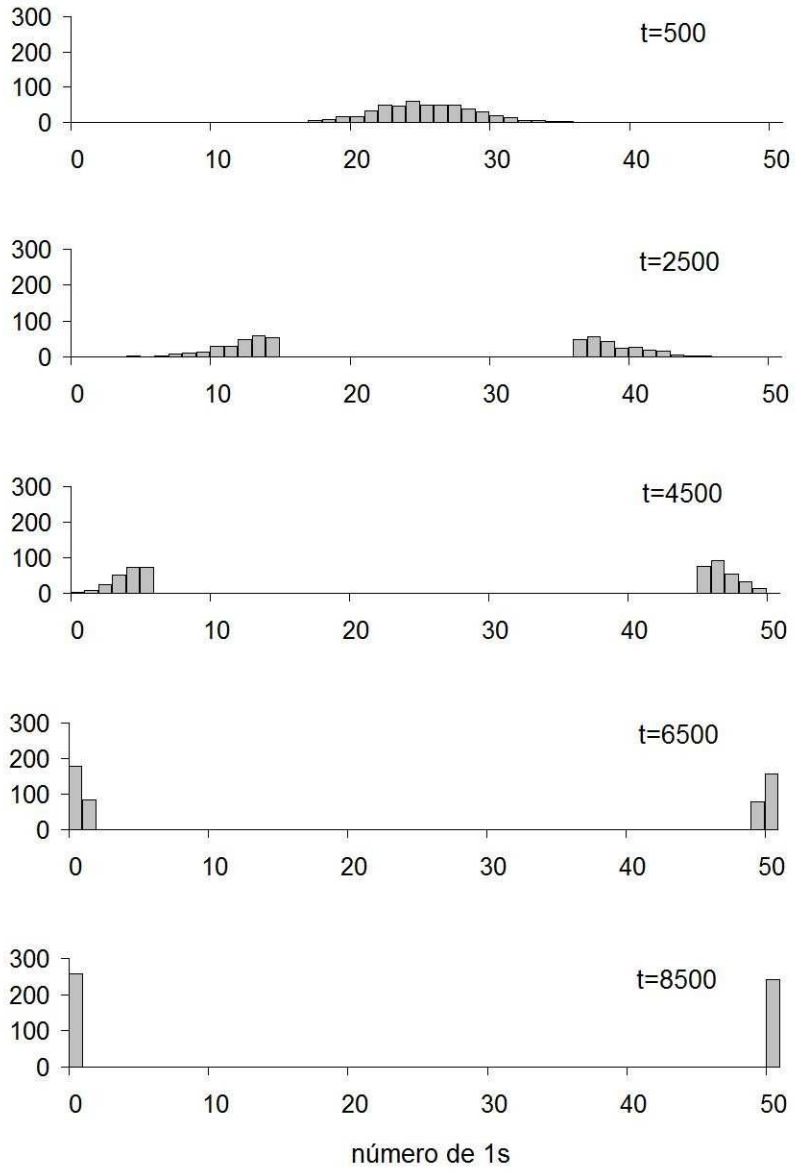


Figura 6.4: Processo evolutivo do φ-PBIL para uma rodada do problema Twomax. Os indivíduos da população estão classificados de acordo com o número de genes em 1 que possuem. O eixo vertical representa a quantidade de indivíduos relacionados a cada classe.

CAPÍTULO 7

CONCLUSÕES E TRABALHOS FUTUROS

Este trabalho propõe e avalia um novo algoritmo de computação evolutiva, da classe dos algoritmos de estimação de distribuição (EDAs). Este novo algoritmo é usado para verificar a hipótese de que o agrupamento da população fornece informações sobre a estrutura do problema e permite a um EDA, mesmo adotando estatísticas de baixa ordem, resolver problemas estruturados que apresentem interação de alta ordem entre as variáveis.

A adoção de algoritmos de agrupamento em computação evolutiva em geral, e em EDAs em particular, não é nova. Duas motivações vinham norteando os trabalhos relatados na literatura até então: (i) garantir a preservação da diversidade da população, prevenindo a convergência para um ótimo local, ou para um único ótimo global, no caso de problemas globalmente multimodais e (ii) evitar a combinação de modelos probabilísticos associados a ótimos globais diferentes, no caso de problemas globalmente multimodais.

Ambas motivações são pertinentes. A manutenção da diversidade permite uma exploração adequada do espaço de busca; sem ela os algoritmos em geral convergem para ótimos locais. Não combinar informações de regiões muito diferentes também é uma motivação razoável, já que freqüentemente os indivíduos distantes estão associados a ótimos locais ou globais que não tem relação entre si e, quando combinados, resultam em regiões inferiores do espaço de busca [Watson e Pollack, 1999].

O trabalho apresentado aqui mostra, porém, que a combinação de informações de regiões diferentes pode, sim, resultar em uma melhor exploracão do espaço de busca. Até mesmo a combinação entre ótimos locais diferentes pode se produtiva, já que estes ótimos podem estar associados à ocorrência de subestruturas complementares que, combinadas, interagem e contribuem para um aumento na função de *fitness* superior à

soma das contribuições individuais.

A relação entre o aprendizado sobre a estrutura do problema e a existência de similaridades entre indivíduos da população absolutamente não é nova nem tampouco desconhecida; pelo contrário, faz parte da fundamentação teórica mais importante em algoritmos genéticos, que é baseada na noção de esquemas [Holland, 1975]. Entretanto, até o momento esta abordagem não havia sido verificada tão diretamente. A diversidade da população vinha sendo compreendida como algo importante a ser mantido, para que outros mecanismos pudessem adequadamente extrair e combinar informações a partir da população. Neste trabalho, a própria diversidade da população é que fornece as informações suficientes sobre as subestruturas do problema.

Foi avaliada uma hipótese segundo a qual o agrupamento da população permitiria capturar a estrutura do problema e com isso seria possível respeitar interações de alta ordem entre as variáveis. Cada grupo, ou subpopulação, estaria inicialmente associado a um esquema, que representa, na teoria de algoritmos genéticos, a ocorrência de similaridades como resultado da existência de subestruturas comuns [Goldberg, 1998].

Um algoritmo é proposto, baseado nestas idéias. φ-PBIL é um algoritmo evolutivo baseado no agrupamento da população, na inferência e utilização de estatísticas de primeira ordem a partir das subpopulações e na adoção de uma medida de informação baseada em estatísticas de segunda ordem, que relacionam cada gene com as possíveis subpopulações. O algoritmo se baseia em um processo computacionalmente simples, que é o agrupamento por meio do algoritmo *k-means*. Os EDAs atualmente capazes de resolver problemas estruturados adotam modelos muito mais complexos e computacionalmente mais caros, principalmente em função da etapa de busca por modelo, que é um problema NP-completo.

Um mecanismo de combinação, fundamental para o funcionamento do algoritmo, utiliza estas estatísticas de segunda ordem para determinar, em cada momento do processo evolutivo, qual é a forma mais adequada de combinar dois modelos de subpopulação. A escolha adotada é por manter as porções mais informativas de cada modelo

"pai" intactas, preservando informações relevantes e interações de alta ordem.

O algoritmo resultante representa uma abordagem bastante parcimoniosa para computação evolutiva, visto que apenas estatísticas de segunda ordem devem ser inferidas e não há a etapa computacionalmente cara de busca por estrutura. A arquitetura incremental é também uma contribuição relevante do ponto de vista conceitual, embora experimentos específicos devam ser conduzidos no futuro para melhor avaliar seus impactos quantitativos, na eficiência e eficácia do algoritmo. Por enquanto, verificou-se apenas que esta arquitetura parece ser tão adequada quanto a alternativa mais convencional, baseada em lotes de indivíduos, os quais são chamados de gerações. Pelo caráter incremental da computação evolutiva, pelo menos conceitualmente a abordagem adotada aqui parece ser adequada.

A avaliação empírica revela a importância do mecanismo de combinação proposto, chamado de combinação guiada por conceito. Sem ele, o algoritmo φ-PBIL não tem capacidade de resolver nenhum problema estruturado, sendo equivalente aos demais EDAs de baixa ordem. Isto sugere que o operador de combinação proposto é, de fato, a contribuição mais relevante deste trabalho, e deve ser explorado em outros contextos de aprendizado artificial e de inferência em geral. Até que ponto o operador de cg-combinação é dependente do restante da estrutura do φ-PBIL é algo que também deve ser verificado.

Os resultados apresentados neste trabalho permitem concluir que (i) o agrupamento da população permite manter no mesmo grupo indivíduos que possuem a mesma subestrutura; (ii) estes grupos estão relacionados ao conceito de esquema, e a pressão de seleção faz com que estes esquemas representem blocos construtores e (iii) a combinação criteriosa e guiada de informaçõ-es sobre grupos diferentes implica na combinação de blocos construtores, o que leva à obtenção da solução ótima global, conforme previsto em [Holland, 1975].

Assim, a hipótese de que o agrupamento da população permite capturar informações sobre a estrutura do problema parece ser válida, de acordo com a verificação empírica

apresentada.

Trabalhos futuros devem incluir uma investigação mais detalhada sobre as propriedades teóricas e assintóticas do algoritmo proposto, tal como é feito usualmente para outros algoritmos de computação evolutiva.

Alguns aspectos do algoritmo podem também ser melhorados no futuro. Para um conjunto de parâmetros, valores *default* foram obtidos por meio de uma metodologia adequada. Entretanto, para outros parâmetros nenhum procedimento mais rigoroso é adotado. Particularmente, deve ser melhor compreendido como ajustar os tamanhos de população para cada tamanho de problema. Algum trabalho a respeito já foi desenvolvido para outros algoritmos [Harik e Lobo, 1999] e a metodologia poderia ser adotada aqui. O número de grupos k também pode ser determinado automaticamente, e as abordagens possíveis para este fim devem ser avaliadas.

Alguns resultados desta tese foram publicados em [Emmendorfer e Pozo, 2007b], [Emmendorfer e Pozo, 2007a], [Emmendorfer e Pozo, 2007c]. Outros resultados deverão ser publicados em [Emmendorfer e Pozo, 2008].

REFERÊNCIAS

[Agresti e Coull, 1998] Agresti, A. e Coull, B. (1998). Approximate is better than exact for interval estimation of binomial proportions. *The American Statistician*, 58:119–126.

[Ahn et al., 2004] Ahn, C., Goldberg, D. E., e Ramakhrishna, R. S. (2004). Multiple-deme parallel estimation of distribution algorithms: basic framework and application. Em *Proceedings of Parallel Processing and Applied Mathematics. Lecture Notes in Computer Science*, páginas 544–551.

[Bacao et al., 2005] Bacao, F., Lobo, V. S., e Painho, M. (2005). Self-organizing maps as substitutes for k-means clustering. Em *International Conference on Computational Science (3). Proceedings*, páginas 476–483.

[Baluja e Caruana, 1995] Baluja, S. e Caruana, R. (1995). Removing the genetics from the standard genetic algorithm. Em *International Conference on Machine Learning*, páginas 38–46.

[Bonet et al., 1997] Bonet, J. S. D., Isbell, C. L., e Viola, P. (1997). MIMIC: Finding optima by estimating probability densities. *Advances in Neural Information Processing Systems*, 9.

[Cavicchio, 1970] Cavicchio, J. (1970). *Adaptive search using simulated evolution.* Doctoral dissertation, University of Michigan, Ann Arbor, MI.

[Chen et al., 2007] Chen, Y.-P., Yu, T.-L., Sastry, K., e Goldberg, D. E. (2007). A survey of linkage learning techniques in genetic and evolutionary algorithms. Relatório Técnico IlliGAL Report No. 2007014, University of Illinois at Urbana-Champaign.

[Chickering, 1996] Chickering, D. M. (1996). Learning bayesian networks is NP-complete. Em *Proceedings of the Fifth International Workshop on Artificial Intelligence and Statistics*, páginas 121–130.

[Coffin e Smith, 2007] Coffin, D. J. e Smith, R. E. (2007). The limitations of distribution sampling for linkage learning. Em *Proceedings of IEEE Congress on Evolutionary Computation (CEC)*.

[Cooper e Herskovits, 1992] Cooper, G. e Herskovits, E. (1992). A bayesian method for the induction of probabilistic networks from data. *Machine Learning*, 9(4):309–347.

[Emmendorfer e Pozo, 2007a] Emmendorfer, L. e Pozo, A. T. R. (2007a). An empirical evaluation of linkage learning strategies for multimodal optimization. Em *Proceedings of IEEE Congress on Evolutionary Computation (CEC)*.

[Emmendorfer e Pozo, 2007b] Emmendorfer, L. e Pozo, A. T. R. (2007b). An incremental approach for niching and building block detection via clustering. Em *Proceedings of the Seventh International Conference on Intelligent Systems Design and Applications*.

[Emmendorfer e Pozo, 2007c] Emmendorfer, L. e Pozo, A. T. R. (2007c). Otimização multimodal através de computação evolutiva e análise de agrupamentos. Em *Encontro Nacional de Inteligencia Artificial, XXVII Congresso da SBC*.

[Emmendorfer e Pozo, 2008] Emmendorfer, L. e Pozo, A. T. R. (2008). Effective linkage learning using low-order statistics and clustering. *IEEE Transactions on Evolutionary Computation*. Submetido.

[Etxeberria e Larrañaga, 1999] Etxeberria, R. e Larrañaga, P. (1999). Global optimization using bayesian networks. Em *Second Symposium on Artificial Intelligence (CIMAF-99)*, páginas 332–339.

[Friedman et al., 1998] Friedman, N., goldszmidt, M., e Koller, D. (1998). The bayesian structural em algorithm. Em *Proceedings of the Fourteenth Conference on Uncertainty in Artificial Intelligence*, páginas 129–138.

[Gama et al., 2005] Gama, J., Medas, P., e Rodrigues, P. (2005). Learning decision trees from dynamic data streams. Em *2005 ACM Symposium on Applierd Computing*, páginas 573–577.

[Goldberg, 1998] Goldberg, D. E. (1998). The race, the hurdle, and the sweet spot: Lessons from genetic algorithms for the automation of design innovation and creativity. Relatório Técnico ILLIGAL Report No. 98007, University of Illinois at Urbana-Champaign.

[Goldberg et al., 1992] Goldberg, D. E., Deb, K., e Clark, J. H. (1992). Genetic algorithms, noise and the sizing of the populations. *Complex Systems*, 6:333–362.

[Goldberg et al., 1989] Goldberg, D. E., Korb, G., e Deb, G. (1989). Messy genetic algorithms: Motivation, analysis and first results. *Complex Systems*, 3:493–530.

[Goldberg e Richardson, 1987] Goldberg, D. E. e Richardson, J. J. (1987). Genetic algorithms with sharing for multimodal function optimization. Em *Proceedings of the Second International Conference on Genetic Algorithms*, páginas 41–49.

[Harik e Lobo, 1999] Harik, G. e Lobo, F. (1999). A parameter-less genetic algorithm. Em *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO-99)*.

[Harik, 1995] Harik, G. R. (1995). Finding multimodal solutions using restricted tournament selection. Em *Proceedings of the International Conference on Genetic Algorithms*, páginas 24–31.

[Harik, 1997] Harik, G. R. (1997). *Learning gene linkage to efficiently solve problems of bounded difficulty using genetic algorithms*. Tese de Doutorado, University of Michigan.

[Harik, 1999] Harik, G. R. (1999). Linkage learning via probabilistic modeling in the ECGA. Relatório Técnico ILLIGAL Report No. 99010, University of Illinois at Urbana-Champaign, Urbana IL.

[Harik et al., 1999] Harik, G. R., Lobo, F. G., e Goldberg, D. E. (1999). The compact genetic algorithm. *IEEE Transactions on Evolutionary Computation*, 3(4):287.

[Holland, 1975] Holland, J. H. (1975). *Adaptation in natural and artificial systems*. University of Michigan Press, Ann Arbor, MI.

[Jakulin e Bratko, 2004] Jakulin, A. e Bratko, I. (2004). Testing the significance of attribute interactions. Em *ICML '04: Proceedings of the twenty-first international conference on Machine learning*, página 52, New York, NY, USA. ACM Press.

[Jong, 1975] Jong, K. A. D. (1975). *AAn Analysis of the Behavior of a Class of Genetic Adaptive Systems*. Doctoral dissertation, University of Michigan.

[Kullback e Leibler, 1951] Kullback, S. e Leibler, R. A. (1951). On information and sufficiency. *Annals of Math. Stats.*, 22:78–86.

[Larrañaga et al., 1999] Larrañaga, P., Etxeberria, R., Lozano, J. A., e Peña, J. M. (1999). Optimization by learning and simulation of bayesian and gaussian networks. Relatório Técnico EHU-KZAA-IK-4/99, Conostia-San Sebastian, Spain: University of the Basque Country.

[Li et al., 2002] Li, J.-P., Balazs, M. E., Parks, G. T., e Clarkson, P. J. (2002). A species conserving genetic algorithm for multimodal function optimization. *Evolutionary Computation*, 10(3):207–234.

[Lima et al., 2005] Lima, C. F., Sastry, K., Goldberg, D. E., e Lobo, F. G. (2005). Combining competent crossover and mutation operators: a probabilistic model building approach. Relatório Técnico ILLIGAL Report No. 2005002, University of Illinois at Urbana-Champaign.

[Llorà e Goldberg, 2003] Llorà, X. e Goldberg, D. E. (2003). Wise breeding GA via machine learning techniques for function optimization. Em *GECCO 2003: Genetic and Evolutionary Computation Conference*, páginas 1172–1183.

[Mahfoud, 1995] Mahfoud, S. W. (1995). *Niching methods for genetic algorithms*. Doctoral dissertation, University of Illinois at Urbana-Champaign,, Urbana, USA.

[McQueen, 1967] McQueen, J. (1967). Some methods for classification and analysis of multivariate observations. Em *Proceedings of the fifth Berkley symposium on Mathematics, Statistics and Probability*, páginas 281–296.

[Michalski, 2000] Michalski, R. S. (2000). Learnable evolution model: Evolutionary process guided by machine learning. *Machine Learning*, 38(1):9–40.

[Michalski, 2003] Michalski, R. S. (2003). Inferential theory of learning and inductive databases. Em *UQAM Summer institute in cognitive sciences*.

[Minsky e Papert, 1969] Minsky, M. e Papert, S. (1969). *Perceptrons: An Introduction to Computational Geometry*. MIT Press, Cambridge, Mass.

[Miquélez et al., 2004] Miquélez, T., Bengoetxea, E., e naga, P. L. (2004). Evolutionary computation based on Bayesian classifiers. *International Journal of Applied Mathematics and Compututer Sciences*, 14(3):335–349.

[Monard e Baranauskas, 2003] Monard, M. e Baranauskas, J. (2003). *Conceitos Sobre Aprendizado de Máquina. In: Sistemas Inteligentes Fundamentos e Aplicações*, volume 1.

[Muhlenbein e Paaβ, 1996] Muhlenbein, H. e Paaβ, G. (1996). From recombination of genes to the estimation of distributions: I binary parameters. Em *Parallel Problem Solving from Nature III*, páginas 178–187.

[Pearl, 1988] Pearl, J. (1988). *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA.

[Pelikan, 2005] Pelikan, M. (2005). *Hierarchical Bayesian Optimization Algorithm: Toward a New Generation of Evolutionary Algorithms*. Springer-Verlag.

[Pelikan e Goldberg, 2000] Pelikan, M. e Goldberg, D. E. (2000). Genetic algorithms, clustering, and the breaking of symmetry. Em *Parallel Problem Solving from Nature VI*, páginas 385–394.

[Pelikan et al., 1999] Pelikan, M., Goldberg, D. E., e E.Cantu-Paz (1999). BOA: The Bayesian optimization algorithm. Em *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO-99)*, páginas 525–532.

[Pelikan et al., 2005a] Pelikan, M., Saltry, K., e Goldberg, D. E. (2005a). Sporadic model building for for efficiency enhancement of hBOA. Relatório Técnico ILLIGAL Report No. 2005026, University of Illinois at Urbana-Champaign, Urbana IL.

[Pelikan et al., 2005b] Pelikan, M., Sastry, K., e Goldberg, D. E. (2005b). Multiobjective hboa, clustering, and scalability. Em *GECCO '05: Proceedings of the 2005 conference on Genetic and evolutionary computation*, páginas 663–670, New York, NY, USA. ACM Press.

[Peña et al., 2005] Peña, J., Lozano, J., e Larrañaga, P. (2005). Globally multimodal problem optimization via an estimation of distribution algorithm based on unsupervised learning of bayesian networks. *Evolutionary Compututation*, 13(1):43–66.

[Petrowski, 1997] Petrowski, A. (1997). A new selection operator dedicated to speciation. Em *Proceedings of the 7th International Converence on Genetic Algorithms*, páginas 144–451.

[Sastry et al., 2005] Sastry, K., Abbass, H. A., Goldberg, D. E., e Johnson, D. D. (2005). Sub-structural niching in estimation of distribution algorithms. Em *GECCO '05: Proceedings of the 2005 conference on Genetic and evolutionary computation*, páginas 671–678, New York, NY, USA. ACM Press.

[Watson et al., 1998] Watson, R. A., Hornby, G., e Pollack, J. B. (1998). Modeling building-block interdependency. Em *PPSN V: Proceedings of the 5th International Con-*

ference on Parallel Problem Solving from Nature, páginas 97–108, London, UK. Springer-Verlag.

[Watson e Pollack, 1999] Watson, R. A. e Pollack, L. B. (1999). Incremental commitment in genetic algorithms. Em *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO-99)*.

[Yu et al., 2005] Yu, T.-L., Sastry, K., , Goldberg, D. E., e Johnson, D. D. (2005). Linkage learning, overlapping building blocks, and systematic strategy for recombination. Em *GECCO '05: Proceedings of the 2005 conference on Genetic and evolutionary computation*, páginas 1217–1224, New York, NY, USA. ACM Press.

[Yuan e Gallagher, 2005] Yuan, B. e Gallagher, M. (2005). On the importance of diversity maintenance in estimation of distribution algorithms. Em *GECCO '05: Proceedings of the 2005 conference on Genetic and evolutionary computation*, páginas 719–726, New York, NY, USA. ACM Press.

[Zang e Poole, 1999] Zang, N. e Poole, D. (1999). On the role of context-specific independence in probabilistic inference. Em *Proceedings of the Sixteenth International Joint Conference on Artificial Intelligence*, páginas 1288–1293.

Livros Grátis

(http://www.livrosgratis.com.br)

Milhares de Livros para Download:

Baixar livros de Literatura
Baixar livros de Literatura de Cordel
Baixar livros de Literatura Infantil
Baixar livros de Matemática
Baixar livros de Medicina
Baixar livros de Medicina Veterinária
Baixar livros de Meio Ambiente
Baixar livros de Meteorologia
Baixar Monografias e TCC
Baixar livros Multidisciplinar
Baixar livros de Música
Baixar livros de Psicologia
Baixar livros de Química
Baixar livros de Saúde Coletiva
Baixar livros de Serviço Social
Baixar livros de Sociologia
Baixar livros de Teologia
Baixar livros de Trabalho
Baixar livros de Turismo