Algoritmos Evolucionários Híbridos

Capítulo 10

Conteúdo

- Motivações para hibridização de AEs.
- Introdução aos algoritmos meméticos.
- Busca local:
  - Adaptação de Lamarck vs. de Baldwin.
- Estrutura de um algoritmo memético.
- Questões de design para algoritmos meméticos:
  - Diversidade, escolha de operadores e uso de conhecimento.
- Exemplo de aplicação:
  - Horário memético de estágios múltiplos
- Referências bibliográficas.

Motivações para Hibridização de AEs

- Modelos cujos AEs se caracterizam por
  - Serem parte de um sistema maior ou
  - Terem outros métodos ou estruturas de dados incorporados a eles.
- Objetivos da hibridização:
  - Modelos híbridos para melhorar desempenho de técnicas já existentes.
  - Modelos híbridos para melhorar a busca por boas soluções.

Motivações para Hibridização de AEs

- Muitos problemas complexos podem ser decompostos em sub-problemas a serem resolvidos com técnicas distintas.
- Não existe um método de propósito geral totalmente bem sucedido ou eficiente.
  - Pode-se combinar AEs com heurísticas específicas.
  - Deve-se ter cautela para não fazer a busca local inviabilizar geração de novas soluções.
  - Busca-se solução híbrida ser melhor que cada um dos algoritmos que geram o modelo híbrido.

Motivações para Hibridização de AEs

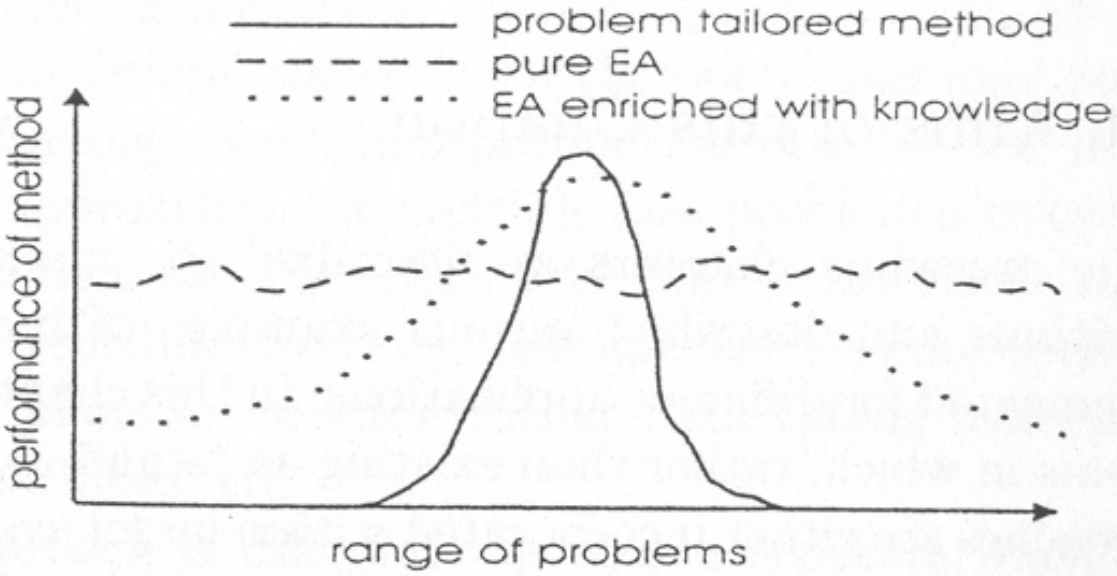


Fig. 10.1. 1990s view of EA performance after Michalewicz [271]

- A quantidade de conhecimento específico no algoritmo híbrido é variável e pode ser ajustada.

Algoritmos Meméticos

- AEs tendem a explorar melhor que explotar:
  - Em parte devido à natureza aleatória dos operadores de variação.
- Pode-se adicionar uma etapa de busca local para melhorar explotação.
- A combinação de AEs com operadores de busca local que atuam dentro do laço do AE é chamada de Algoritmo Memético.
  - O termo memético também é usado para AEs cujos operadores empregam conhecimento específico de instâncias do problema.

Algoritmos Meméticos

- Idéia dos mimes (Dawkin):

  Memes – Unidades de transmissão cultural.

  Genes – Unidades de transmissão biológica.
- A interação meme-gene pode inserir aprendizagem ao ciclo evolucionário (desenvolvimento). Assim, um meme pode transformar um candidato a solução.
- Algoritmos Meméticos podem ser muito mais rápidos e mais precisos que AEs em alguns problemas.

Busca Local

- Busca local é um processo iterativo para examinar o conjunto de pontos na vizinhança da solução atual e a substituir por um vizinho melhor, caso exista.

```
BEGIN
  /* given a starting solution i and a neighbourhood function n */
  set best = i;
  set iterations = 0;
  REPEAT UNTIL ( depth condition is satisfied ) DO
    set count = 0;
    REPEAT UNTIL ( pivot rule is satisfied ) DO
      generate the next neighbour j ∈ n(i);
      set count = count + 1;
      IF (f(j) is better than f(best)) THEN
        set best = j;
      FI
    OD
    set i = best;
    set iterations = iterations + 1;
  OD
END
```

Busca Local

- Componentes principais que afetam o algoritmo:
- Regra de pivoteamento (encerra o laço interno):
  - Ascendente mais íngreme (*steepest ascent*): Busca entre todos vizinhos aquele mais apto.

    count = |n(i)| - busca em toda vizinhança.
  - Ascendente elitista ou primeiro ascendente (*greedy ascent* ou *first ascent*): Busca o primeiro vizinho que se mostrar mais rápido.

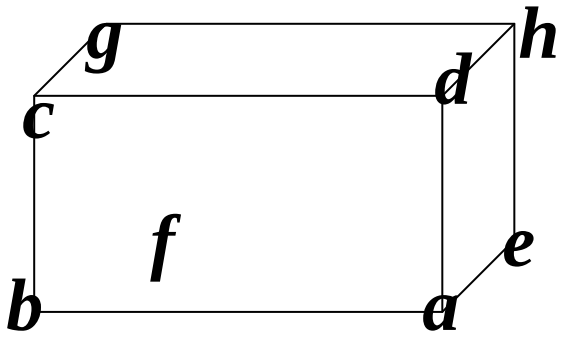
    (count = |n(i)|) or (best != i) – busca primeira melhoria.
- Condição de profundidade (encerra o laço externo):
  - Determina o número de interações da busca local.
- Escolha da função geradora de vizinhança.

Busca Local

- Escolha da função de vizinhança:
  - *N*(*i*) é freqüentemente definida como um conjunto de pontos que podem ser alcançados através da aplicação de algum operador de movimento ao ponto i.
- Representação equivalente: Grafo não direcionado *G(v,E)*:
  - *v* é o conjunto de vértices que representa todos pontos representáveis, as soluções em potencial.
  - *E* é o conjunto de arestas que representa as possíveis transições que podem ocorrer a partir da aplicação de um único operador. As arestas podem ser direcionadas e ponderadas.

Busca Local

- Em resumo:
  - É definida pela combinação de vizinhança, *N(x)*, e regra de pivoteamento.
  - É relacionada à metáfora da paisagem (*landscape*).
  - *N(x)* é definida como o conjunto de pontos que podem ser alcançados a partir da aplicação de um operador de movimento.
  - Exemplo: Busca *bit flipping* em problemas binários.



$$N(d) = \{a,c,h\}$$

Busca Local: Variações

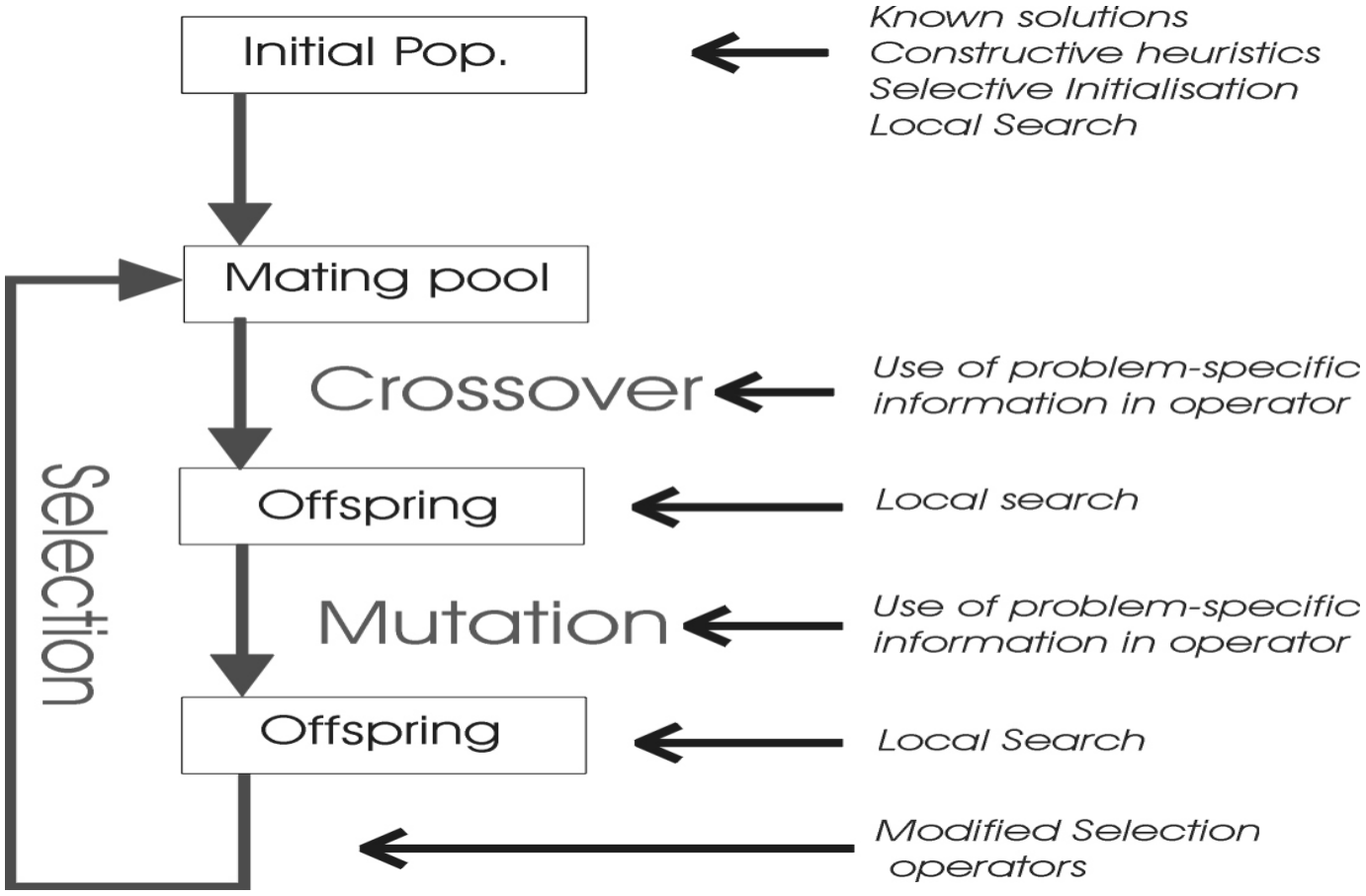
- A busca pode ocorrer no espaço de busca ou no espaço de soluções.
- Ajustar o número de interações a serem realizadas pela busca local.
- A busca local pode ser aplicada para
  - A população toda;
  - O indivíduo mais apto;
  - O individuo menos apto.

Lamarckianismo e Efeito Baldwin

- Herança das mudanças feitas ao indivíduo (características adquiridas):
  - Um AM é dito Lamarckiano se o indivíduo resultante da busca local substitui o original. Isto é, as características adquiridas durante seu tempo de vida são herdadas por sua prole.
  - Um AM é dito Baldwiniano se o processo evolucionário for guiado pelas adaptações favoráveis sem que as modificações nas aptidões dos indivíduos, devido a aprendizagem ou desenvolvimento, se convertam em mudanças de características genéticas.
  - Abordagem Baldwiniana ou uma combinação probabilística dos dois mecanismos.

Estrutura de um Algoritmo Memético

- Possíveis pontos onde se pode hibridizar.



Estrutura de um Algoritmo Memético

Inicialização

- Incialização heurística vs. aleatória.

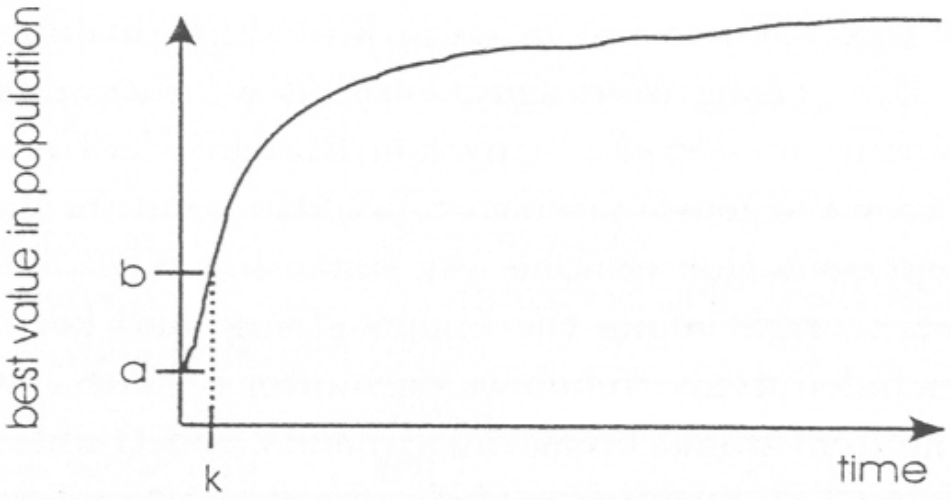


Fig. 2.6. Illustration of why heuristic initialisation might not be worth additional effort. Level a shows the best fitness in a randomly initialised population, level b belongs to heuristic initialisation

Estrutura de um Algoritmo Memético

Inicialização

- Possíveis benefícios por iniciar AEs com soluções existentes:
  - Evitar a perda de esforço computacional refletindo aumento de eficiência (velocidade).
  - Direcionar busca para regiões promissoras do espaço de busca levando a aumentar efetividade (qualidade da solução final).
  - Dividir o esforço computacional entre inicialização heurística e busca evolucionária pode gerar melhores resultados que gastar todo o esforço em busca evolucionária apenas.

Estrutura de um Algoritmo Memético
Inicialização

- Maneiras de mudar a função de inicialização em relação a um critério aleatório:
  - Semeadura (*Seeding*): Semeia a população com uma ou mais soluções boas, vindas de outras técnicas:
    - Exemplos de emprego de informação específica do problema: heurística do vizinho mais próximo para os problemas do tipo TSP e agendamento do mais difícil em primeiro lugar para problemas de agenda e planejamento.
  - Incialização Seletiva (*Selective initialisation*): Cria-se grande número de soluções aleatórias e seleciona-se a população inicial a partir delas:
    - Realizar N torneios de tamanho k;
    - Selecionar baseando-se também na diversidade.

Estrutura de um Algoritmo Memético
Inicialização

- Ainda maneiras de mudar a função de inicialização em relação a um critério aleatório:
  - Realizar busca local em cada membro da população escolhido aleatoriamente. Produz-se um conjunto de indivíduos localmente ótimos.
  - Empregar um ou mais métodos anteriores para identificar uma ou mais boas soluções:
    - Deve clonar e aplicar mutação com altas taxas (*mass mutation*) nas soluções iniciais encontradas de forma a gerar um número de indivíduos na vizinhança do ponto inicial.

Estrutura de um Algoritmo Memético
Operadores Inteligentes

- Operadores inteligentes de variação: Incorporam conhecimento específico da instância ou do problema ao operador.
- Introdução de viés (*bias*) nos operadores:
  - Exemplo 1: Aumentar a probabilidade de escolha de características mais compactas para emprego em um classificador.
  - Exemplo 2: Genes codificando instruções de microprocessador de modo a serem agrupados em conjuntos de efeitos similares.

Estrutura de um Algoritmo Memético
Operadores Inteligentes

- Uso de conhecimento específico do problema:
  - Incorporação de uma fase de busca local dentro de um operador de recombinação.
  - Ex.: Testar todas as possíveis orientações dos dois fragmentos de estrutura de proteína “separados” pelo ponto de cruzamento e pegar a energeticamente mais favorável. Se nenhum ajuste possível for encontrado, escolher outro ponto de cruzamento.
- Uso de conhecimento específico da instância:
  - Operadores recebem heurística muito específica.
  - Ex.: *Merz and Friesleben’s distance-preserving crossover (DPX) operator* para o TSP: Filhos recebem sub-rotas de pais.

Estrutura de um Algoritmo Memético
Operadores Inteligentes

- Busca local atuando sobre todas as soluções criadas pelos operadores de variação.
  - É compatível com o conceito de meme, pois realiza uma ou mais fases de melhorias de indivíduos durante um ciclo do AE.
  - É o tipo mais comum.
- Exemplo:

```
BEGIN
  INITIALISE population;
  EVALUATE each candidate;
  REPEAT UNTIL ( TERMINATION CONDITION is satisfied ) DO
    SELECT parents;
    RECOMBINE to produce offspring;
    MUTATE offspring;
    EVALUATE offspring;
    IMPROVE offspring via Local Search;
    SELECT individuals for next generation;
  OD
END
```

Estrutura de um Algoritmo Memético
Operadores Inteligentes

- Hibridização durante mapeamento do genótipo para fenótipo.
  - Precede a avaliação da aptidão.
  - Ex.: Uso de conhecimento específico da instância dentro do decodificador ou da função de reparo. Seção 2.4.2 – problema da mochila (*knapsack problem*).
  - Para este e demais tipos de operadores inteligentes, o AE permite o uso de heurísticas com menos viés ou divisão do problema. Contudo, as soluções são poucos escalonáveis.

Design para Algoritmos Meméticos

- AMs não são “soluções mágicas” para problemas de otimização.
- Deve-se tomar cuidado com:
  - Diversidade da população.
  - Escolha dos operadores.
  - Uso do conhecimento adquirido durante o processo de otimização.

Design para Algoritmos Meméticos
Preservação de Diversidade

- O problema da convergência prematura.
- Técnicas para combater este problema:
  - Usar somente uma proporção pequena de boas soluções conhecidas na população inicial.
  - Usar operadores de recombinação que são projetados para preservar diversidade (como Merz's DPX).
  - Modificar operador de seleção para prevenir duplicações.
  - Modificar operador de seleção ou o critério de aceitação da busca local para usar um método de Boltzmann.

Design para Algoritmos Meméticos
Preservação de Diversidade

- Manutenção de diversidade para população é considerada em algoritmos com algum sucesso:
  - O cruzamento DPX de Merz explicitamente gera indivíduos que são eqüidistantes para cada pai, tendo a mesma distância entre os pais.
  - Operador adaptativo de Boltzmann, por Krasnogor, usa critério de aceitação de novo indivíduo baseado no cozimento simulado. A temperatura é inversamente proporcional à diversidade da população.

Design para Algoritmos Meméticos
AM de Boltzmann: Critério de Aceitação

- Para um problema de maximização, a probabilidade de se aceitar um novo vizinho é definida abaixo.
  - Seja Δf = aptidão do vizinho – aptidão original
  - *k* é uma constante de normalização.

$$P(\text{accept neighbour}) = \begin{cases} 1 & \Delta f > 0 \\ e^{\frac{k\Delta f}{f_{\max} - f_{avg}}} & \Delta f < 0 \end{cases}$$

Design para Algoritmos Meméticos
AM de Boltzmann: Critério de Aceitação

- A dinâmica induzida é tal que:
  - Se a população é diversa => espalhamento da aptidão (Δf) é alto => aceitação majoritária de movimentos “melhoradores” de aptidão => Explotação.
  - Se a população é convergente (pouco dispersa) => espalhamento da aptidão (Δf) é baixo => aceitação majoritária de movimentos “pioradores” de aptidão => Explotação.
- Krasnogor testou este modelos em problemas tipo TSP e de predição de estruturas de proteínas.

Design para Algoritmos Meméticos
Escolha de Operadores

- O fator crucial no projeto de um AM é:
  - Escolha da heurística de melhoramento ou do operador de movimento da busca local.
    - Merz e Freisleben mostram que a escolha do operador de movimento pode ter efeito dramático na eficiência e na efetividade da busca local.
    - Krasnogor mostrou vantagens em usar busca local com um operador de movimento que é diferente dos operadores de movimento usados pela mutação e pelo cruzamento.
    - Podem-se disponibilizar múltiplos operadores de busca local, escolhendo-se qual a ser aplicado em cada caso.

Design para Algoritmos Meméticos
Uso de Conhecimento

- Como usar e reusar o conhecimento adquirido durante o processo de otimização?
  - Na recombinação.
  - De modo geral não são usados mecanismos explícitos.
  - Uma possível hibridização – *tabu search.*
  - Extensões dos esquemas de aceitação/seleção de Boltzmann.
    - Uso de informação sobre os genomas da população atual e/ou populações anteriores.

Exemplo de Aplicação

- O problema de Agendamento (*Timetabling*):
  - Conjunto de eventos *E* a serem agendados em um conjunto de períodos de tempo *P*.
    - Restrições *hard* e *soft*: Número de lugares necessários, impossibilidade de pessoa estar simultaneamente em 2 locais.
  - Existem diferentes instâncias para este problema.
    - Ex.: Exames que alunos devem fazer em uma universidade.

      |*E*| exames a serem agendados em |*P*| períodos com |*S*| acentos disponíveis para cada período.
  - Matriz de co-ocorrência *C*:
    - c_{ij} = núm. de estudantes que precisam fazer exames *i* e *j*.
  - Problema de otimização com restrições, tratado através do uso de uma função de penalização.

Exemplo de Aplicação

- Função de penalidade considera que:
  - Exames não podem ser agendados em períodos que não possuem número adequado de acentos.
  - É “muito indesejável” que dois exames *i* e *j* sejam agendados para o mesmo período se $c_{ij} > 0$.
  - Não é desejável que os estudantes façam 2 exames no mesmo dia.
  - É preferível que estudantes não façam exames em períodos consecutivos, mesmo que haja uma noite entre eles.

Exemplo de Aplicação

- Características interessantes da abordagem documentada por Burke e Newell:
  - Abordagem de decomposição (em subconjuntos).
  - A heurística de otimização utilizada é um AE.
  - O AE incorpora outras heurísticas (ele é um AM).

Exemplo de Aplicação

- Escalonador heurístico:
  - Um escalonador heurístico divide *E* em um número de subconjuntos menores, de mesmo tamanho, que terão seus elementos agendados (decomposição).
    - Os elementos do subconjunto *n* só são agendados após os componentes dos *n-1* subconjuntos que não se alteram mais.
  - A ordem de agendamento de exames é definida por 3 métricas que estimam a dificuldade de agendar cada exame (heurística de seqüenciamento), considerando:
    - Conflitos com exames ainda não agendados;
    - Conflitos com exames já agendados;
    - Horários válidos disponíveis para um exame.

Exemplo de Aplicação

- Escalonador heurístico (cont):
  - Então os subgrupos são criados conforme o algoritmo de otimização (que irá agendar os elementos do subconjunto) é executado, de forma que o primeiro subconjunto possui os eventos mais difíceis de serem agendados e o último subconjunto possui os eventos mais fáceis de serem agendados.

Exemplo de Aplicação

- O escalonador heurístico pode ser usado com uma das várias técnicas para lidar com agendamento de cada subgrupo. Foi escolhido um AM:

Representation	Set of linked lists of exams. each encoding for one period
Recombination	None
Mutation	Random choice of "light" or "heavy" mutation
Mutation probability	100%
Parent selection	Exponential ranking
Survival selection	Best 50 of 100 offspring
Population size	50
Initialisation	Randomly generated then local search applied to each
Termination condition	Five generations with no improvement in best fitness
Special Features	Local search (to local optimum) applied after mutation

Table 10.1. Table describing MA embedded with multistage timetabling algorithm

Referências

- Eiben, A.E., Smith, J. E.. Hybridisation with Other Techniques: Memetic Algorithms. In Eiben, A.E., Smith, J. E.. **Introduction to Evolutionary Computing**. Berlin: Springer, 2003. p. 173-188.
- Burke, E. K., Newall, J. P. A multi-stage evolutionary algorithm for the timetable problem. **IEEE Transactions on Evolutionary Computation**, v. 3, n. 1, p. 63-74, April 1999.
- B. Frieselben and P. Merz. A genetic local search algorithm for solving symmetric and asymmetric travelling salesman problems. **Proceedings of the 3rd IEEE International Conference of Evolutionary Computing**, p. 616-621, May 1996.