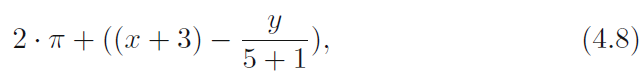
Conforme explicado no Capítulo 3, existem duas forças fundamentais que formam a base dos sistemas evolucionários: variação e seleção. Neste capítulo, discutimos os componentes do EA por trás do primeiro. Como os operadores de variação trabalham no equivalente ao nível genético, ou seja, trabalham na representação de soluções, e não nas próprias soluções, este capítulo é subdividido em seções que tratam de diferentes maneiras pelas quais as soluções podem ser representadas e variadas dentro do algoritmo de pesquisa geral.

**4.6 Representação da Árvore**

Árvores estão entre as estruturas mais gerais de representação de objetos na computação e formam a base para o ramo dos algoritmos evolutivos conhecido como programação genética (GP). Em geral, (analisar) árvores capturam expressões em uma determinada sintaxe formal. Dependendo do problema em questão e das percepções dos usuários sobre como as soluções devem ser, isso pode ser a sintaxe de expressões aritméticas, fórmulas em lógica de predicado de primeira ordem ou código escrito em uma linguagem de programação. Para ilustrar o problema, vamos considerar um de cada um destes tipos de expressões.

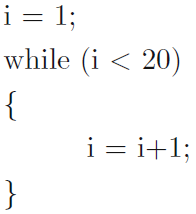
uma fórmula aritmética:



uma fórmula lógica:

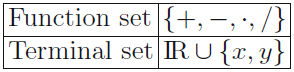


o seguinte programa:

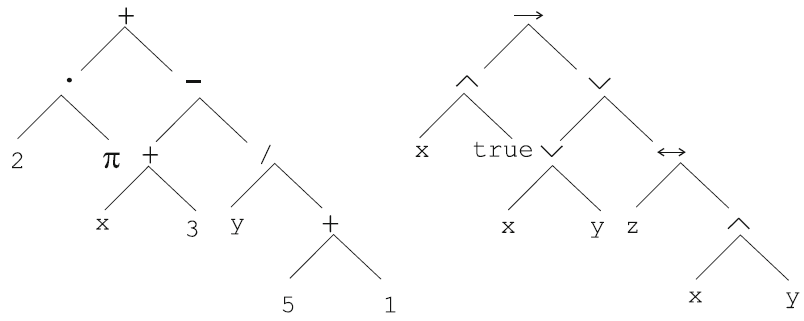


As Figuras 4.18 e 4.19 mostram as árvores de análise pertencentes a essas expressões. Esses exemplos ilustram geralmente como as árvores de análise podem ser usadas e interpretadas.

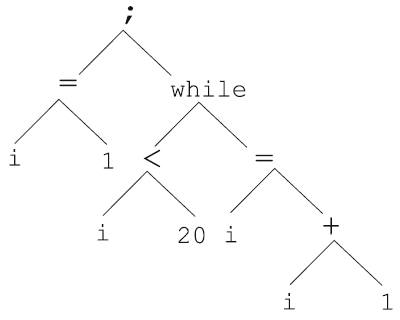
Tecnicamente falando, a especificação de como representar indivíduos se resume em definir a sintaxe das árvores, ou de forma equivalente, a sintaxe das expressões simbólicas (**s-expressões**) que elas representam. Normalmente, isso é feito definindo um **conjunto de funções** e um **conjunto de terminais**. Os elementos do conjunto de terminais são permitidos como folhas, enquanto os símbolos do conjunto de funções são nós internos. Por exemplo, uma função adequada e um conjunto de terminais que permitem a expressão na Eq (4.8) como sintaticamente correta são fornecidos na Tabela 4.4.



**Tabela 4.4**. Função e conjunto de terminais que permitem a expressão na Eq. (4.8) como sintaticamente correto



**Fig 4.18** Analisar árvores pertencentes às Eqs (4.8) (esquerda) e (4.9) (direita)



**Fig. 4.19** Analisar a árvore pertencente ao programa acima

Estritamente falando, devemos especificar a aridade (o número de atributos necessários) para cada símbolo de função no conjunto de funções, mas para funções lógicas ou aritméticas padrão isso é frequentemente omitido. Da mesma forma, uma definição de expressões corretas (árvores) com base na função e no conjunto de terminais deve ser fornecida. No entanto, como segue a maneira geral de definir termos em linguagens formais, também é frequentemente omitido. Para fins de integridade, fornecemos abaixo:

* Todos os elementos do conjunto de terminais T são expressões corretas.
* Se f ∈ F é um símbolo de função com aridade n e e1,. . . , en são expressões corretas, então f (e1,..., en) também é.
* Não existem outras formas de expressões corretas.

Observe que, nesta definição, não distinguimos diferentes tipos de expressões; cada símbolo de função pode receber qualquer expressão como argumento. Esse recurso é conhecido como **propriedade de fechamento**.

Na prática, os símbolos de função e os símbolos terminais são freqüentemente digitados e impõem requisitos sintáticos extras. Por exemplo, pode-se precisar de símbolos de função aritmética e lógica, por exemplo, para permitir (N = 2) ∧ (S> 80.000)) como uma expressão correta. Nesse caso, é necessário garantir que um símbolo de função aritmética (lógica) tenha apenas argumentos aritméticos (lógicos), por exemplo, para excluir N ∧ 80.000 como uma expressão correta. Este problema é abordado na programação genética fortemente tipada [304].

**4.6.1 Mutação para representação de árvore**

A implementação mais comum de mutação baseada em árvore funciona selecionando um nó aleatoriamente da árvore e substituindo a subárvore começando ali por uma árvore gerada aleatoriamente. Essa subárvore recém-criada geralmente é gerada da mesma maneira que na população inicial (Seção 6.4) e, portanto, está sujeita às condições de profundidade e largura máximas. A Figura 4.20 ilustra como a árvore de análise pertencente à Eq. (4.8) (à esquerda) é transformado em um que representa 2 · π + ((x + 3) - y). Observe que, uma vez que um nó é selecionado aleatoriamente para ser o ponto de substituição, e conforme um desce através de uma árvore, existem potencialmente mais nós em qualquer profundidade, o tamanho (profundidade da árvore) do filho pode exceder o da árvore pai.

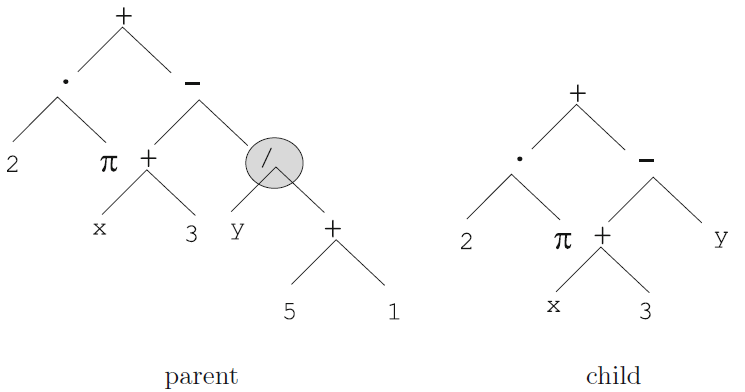


Fig. 4.20. Mutação baseada em árvore ilustrada: o nó designado por um círculo na árvore à esquerda é selecionado para mutação. A subárvore que começa nesse nó é substituída por uma árvore gerada aleatoriamente, que é uma folha aqui

A mutação baseada em árvore tem dois parâmetros:

* a probabilidade de escolher a mutação na junção com a recombinação
* a probabilidade de escolher um ponto interno dentro do pai como a raiz da subárvore a ser substituída

É notável que o livro clássico de Koza sobre GP de 1992 [252] aconselha os usuários a definir a taxa de mutação em 0, ou seja, sugere que a GP funciona sem mutação. Mais recentemente, Banzhaf et al. recomendado 5% [37]. Ao dar à mutação um papel tão limitado, o GP difere de outros fluxos de EA. A razão para isso é a visão geralmente compartilhada de que o crossover tem um grande efeito de embaralhamento, agindo em certo sentido como um operador de macromutação [9]. A prática de GP atual usa frequências de mutação baixas, mas positivas, embora alguns estudos indiquem que a sabedoria comum que favorece uma abordagem de cruzamento (quase) puro pode ser enganosa [275].

**4.6.2 Recombinação para Representação em Árvore**

A recombinação baseada em árvore cria descendentes por meio da troca de material genético entre os pais selecionados. Em termos técnicos, é um operador binário que cria duas árvores filhas a partir de duas árvores pais. A implementação mais comum é o cruzamento de subárvore, que funciona trocando as subárvores começando em dois nós selecionados aleatoriamente nos pais fornecidos. Isso é ilustrado na Fig. 4.21. Observe que o tamanho (profundidade da árvore) dos filhos pode exceder o das árvores pais. Nesse sentido, a recombinação dentro do GP difere da recombinação em outros dialetos da CE. A recombinação baseada em árvore tem dois parâmetros:

* a probabilidade de escolher a recombinação na junção com a mutação
* a probabilidade de escolher nós internos como pontos de cruzamento

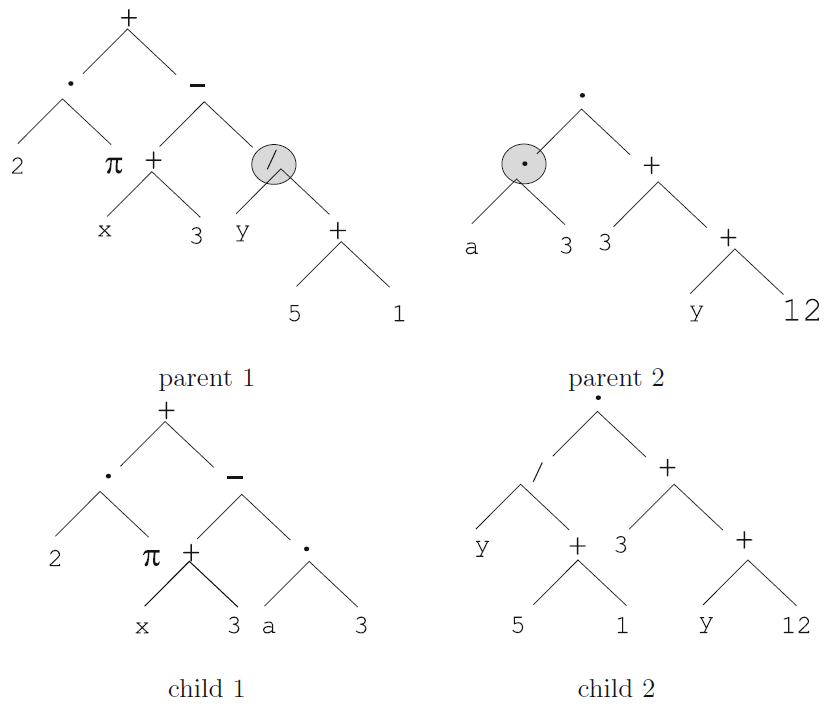


Fig. 4.21. Cruzamento baseado em árvore ilustrado: os nós designados por um círculo nas árvores-pai são selecionados para servir como pontos de cruzamento. As subárvores que olham para esses nós são trocadas, resultando em duas novas árvores, que são as filhas