Algoritmo DPLL

O **algoritmo DPLL/Davis-Putnam-Logemann-Loveland** é um completo algoritmo baseado em backtracking(re-leitura ou voltar atrás) para decidir a satisfatibilidade das fórmulas de lógica proposicional na forma normal clausal, isto é, para solucionar o problema SAT.

O algoritmo foi introduzido em 1962 por Martin Davis, Hilary Putnam, George Logemann e Donald W. Loveland, ele é um refinamento do algoritmo de Davis-Putnam mais antigo, o qual é baseado num processo de resolução desenvolvido por Davis e Putnam em 1960. Principalmente em publicações antigas, o algoritmo de Davis-Logemann-Loveland é freqüentemente referenciado como "O Método Davis-Putnam" ou o "Algoritmo DP". Outros nomes comuns que mantém a distinção são DLL e DPLL.

DPLL é um procedimento altamente eficiente, e forma a base para os mais eficientes solucionadores SAT e outros problemas NP-completos que podem ser reduzidos para o problema SAT, e também para muitos provadores de teoremas para os fragmentos da lógica de primeira ordem.

O Algoritmo

A idéia básica do algoritmo é a de construir uma valoração para uma fórmula fornecida como um conjunto de cláusulas. O algoritmo funciona, inicialmente, selecionando um literal qualquer de uma das cláusulas da fórmula, atribuindo-lhe um valor de verdade. Logo após, simplifica-se a fórmula e então verifica recursivamente se a fórmula simplificada é satisfatível. Se este for o caso, a fórmula original também é satisfatível; do contrário, a mesma verificação recursiva é feita, assumindo agora o valor de verdade oposto. Isto é conhecido como *regra de divisão*(*splitting rule*), que divide o problema em dois sub-problemas mais simples. O passo de simplificação, em sua essência, remove todas as cláusulas que se tornam verdade de acordo com a assinatura da fórmula, ou seja, as cláusulas que se tornam verdade a partir da atribuição do valor verdade ao literal escolhido, neste passo, removem-se também todos os literais opostos das cláusulas remanescentes.

O algoritmo DPLL faz o uso das regras abaixo no passo de simplificação:

Propagação Unitária

> Se uma cláusula for uma *cláusula unitária*, isto é, contém somente um único literal, esta cláusula pode ser satisfeita somente atribuindo o valor necessário para fazer este literal se tornar verdadeiro. Na prática, isto leva à atribuição do valor verdade a outras cláusulas que também possuem este literal, diminuindo assim o espaço de busca.

Eliminação de Literais Puros

> Se um literal ocorrer somente com uma polaridade na fórmula ele será chamado *literal puro*. Os literais puros podem sempre ser atribuídos de modo que faça todas as cláusulas que os contêm se tornarem verdadeiras. Assim, estas cláusulas podem ser suprimidas na busca. Esta otimização é parte do algoritmo original de DPLL, mas algumas implementações omitem-na, ou porque o efeito em execuções eficientes é insignificante ou devido ao custo para detectar um literal puro.

A insatisfatibilidade de uma atribuição parcial dada é detectada quando uma cláusula se torna vazia, ou seja, quando uma cláusula tem todos seus literais falsificados através da atribuição de valores verdade aos literais opostos. A insatisfatibilidade completa da fórmula somente pode ser detectada após a busca exaustiva, ou seja, se todas as atribuições parciais possíveis forem insatisfatíveis. A satisfatibilidade da fórmula é detectada quando todas suas variáveis são atribuídas de forma que não gere nenhuma *cláusula vazia*, em algumas implementações modernas, ocorre quando todas as cláusulas forem satisfeitas.

Pseudocódigo

O algoritmo DPLL pode ser sumarizado no seguinte pseudocódigo, onde Φ é a fórmula na FNC e μ uma atribuição parcial de verdade que está inicialmente vazia:

```
função DPLL(Φ, μ)
   se todas as cláusulas de Φ forem verdadeiras
       então retorne verdadeiro;
   se alguma cláusula de Φ for falsa
       então retorne falso;
   se ocorrer uma cláusula unitária c em Φ
       então retorne DPLL(atribuição(c,Φ), μ ∧ c);
   se ocorrer um literal puro c em Φ
       então retorne DPLL(atribuição(c,Φ), μ ∧ c);
   c := escolha_literal(Φ);
   retorne DPLL(atribuição(c,Φ), μ ∧ c) ou DPLL(atribuição(¬c,Φ), μ ∧
¬c);
```

No pseudocódigo acima, atribuição(c, Φ) é uma função que retorna uma fórmula obtida pela substituição de cada ocorrência de c por verdadeiro, e cada ocorrência do literal oposto por falso na fórmula Φ, e em seguida, simplificando a fórmula resultante. A função DPLL do pseudocódigo retorna verdadeiro se a atribuição final satisfaz a fórmula ou falso se tal atribuição não satisfaz a fórmula. Em uma implementação real, a atribuição satisfatível também é retornada no caso de sucesso (esta foi omitida para maior clareza).

Outras Considerações

O **algoritmo de Davis-Logemann-Loveland depende** da escolha do *literal ramificado*, que é o literal considerado na etapa de backtracking. Em conseqüência, este não é exatamente um algoritmo, mas sim uma família de algoritmos, um para cada maneira possível de escolher o literal ramificado. A eficiência é fortemente afetada pela escolha do literal, existem exemplos no qual o tempo de execução é constante ou exponencial dependendo da escolha dos literais ramificados.

Ver também

- Problema de satisfatibilidade booleana
- Algoritmo de Davis-Putnam
- Algoritmo Chaff
- Princípio da Resolução

Referências

- SILVA, F. C. ; FINGER, M. ; MELO, A. C. V.. *Lógica para Computação* [1] .São Paulo: Thomson Learning, 2006. ISBN 85-221-0517-0

Referências

[1] http://www.thomsonlearning.com.br/detalheLivro.do?id=104261

Fontes e Editores da Página

Algoritmo DPLL *Fonte*: http://pt.wikipedia.org/w/index.php?oldid=19352506 *Contribuidores*: Celopinho, Gunnex, Ivsmjunior, Rauljcs, Salgueiro, 2 edições anónimas

Licença