SAT & Sudoku

Ruben Martins

Instituto Superior Técnico

Lisboa, 26 de Abril de 2006

Resumo

- Introdução ao Mundo SAT,
- Algoritmos SAT,
- Exemplos de Aplicação,
- Conclusões.

Lógica Proposicional

- Domínio das variáveis: 1 (Verdadeiro) ou 0 (Falso),
- Operações Lógicas: $\vee$ (ou), $\wedge$ (e), $\neg$ (negação),
  - De notar que $\alpha \rightarrow \beta$ é equivalente a $\neg\alpha \vee \beta$.
- Se α e β forem variáveis Booleanas então teremos por exemplo as seguintes fórmulas proposicionais:
  - $\alpha \wedge \beta$,
  - $\alpha \vee 1 = 1$,
  - $\beta \wedge 0 = 0$.
- As seguintes fórmulas não são fórmulas proposicionais:
  - $3 + x = x + 3$,
  - $\forall_a \exists_b (a \vee b) \wedge (\neg a \vee \neg b)$.

O que é SAT?

Problema SAT

O problema SAT consiste em, dada uma fórmula Booleana, determinar se existe ou não uma atribuição de valorações que tornem a fórmula verdadeira.

- Tem como base a Lógica Proposicional,
- Parece fácil no entanto a dificuldade aumenta rapidamente conforme o Problema cresce em tamanho,
- O tamanho da fórmula é medido no número de variáveis e no número de operações que esta contém.

Forma Normal Conjuctiva (CNF)

Forma Normal Conjuctiva (CNF)

Uma fórmula proposicional está na sua forma CNF quando é a conjunção de cláusulas. Uma cláusula é a a disjunção de literais. Um literal é a variável ou a sua negação.

Exemplo

$(\neg\alpha \vee \beta) \wedge (\alpha \vee \neg\gamma \vee \neg\beta) \wedge (\neg\alpha \vee \gamma)$

Porquê é que SAT é importante?

- Importância Teórica:
  - Primeiro problema NP-Completo, descoberto por Cook em 1971.

Problema NP-Completo

- Problema NP,
- Problema NP-Díficil, ou seja, todos os outros problemas em NP se reduzem a ele.

Porquê é que SAT é importante?

- Importância Práctica:
  - Resolução de Problemas Combinatórios,
  - Verificação e Modelação de Software,
  - Verificação Sequencial de Circuitos,
  - Aplicações à Biocomputação,
  - ...
- Existência de poderosos algoritmos SAT capazes de resolver problemas reais de modo eficiente.

Introdução

- Davis, Putnam, 1960:
  - Baseado no método da resolução,
- Davis, Logemann, Loveland, 1962:
  - Baseado na pesquisa,
  - Base dos algoritmos mais recentes,
  - Aprendizagem e retrocesso não cronológico, 1996.
- Uso de técnicas SAT para outro tipo de restrições.

Davis, Putnam, 1960

- Davis, Putnam, 1960:
  - Baseado no método da resolução,
- Davis, Logemann, Loveland, 1962:
  - Baseado na pesquisa,
  - Base dos algoritmos mais recentes,
  - Aprendizagem e retrocesso não cronológico, 1996.
- Uso de técnicas SAT para outro tipo de restrições.

Algoritmo de Davis Putnam

Resolução

O método de resolução consiste na escolha de uma variável de forma a que esta seja incompatível em duas cláusulas alvo.

Exemplo

$(\alpha \vee \beta \vee \sigma) \wedge (\gamma \vee \neg\beta \vee \alpha)$

Algoritmo de Davis Putnam

Resolução

O método de resolução consiste na escolha de uma variável de forma a que esta seja incompatível em duas cláusulas alvo.

Exemplo

$(\alpha \vee \beta \vee \sigma) \wedge (\gamma \vee \neg\beta \vee \alpha)$

Algoritmo de Davis Putnam

Resolução

O método de resolução consiste na escolha de uma variável de forma a que esta seja incompatível em duas cláusulas alvo.

Exemplo

$(\alpha \vee \beta \vee \sigma) \wedge (\gamma \vee \neg\beta \vee \alpha)$
$(\alpha \vee \sigma \vee \gamma)$

Algoritmo de Davis Putnam

- Escolher de modo iterativo variáveis para aplicar o método de resolução até não existirem mais variáveis,
- Podemos descartar as cláusulas originais após cada iteração,
- Se no fim do método todos os literais forem puros então é devolvido SAT.

Exemplo SAT

$$(\alpha \vee \beta \vee \sigma) \wedge (\beta \vee \neg\sigma) \wedge (\neg\beta \vee \gamma)$$

Algoritmo de Davis Putnam

- Escolher de modo iterativo variáveis para aplicar o método de resolução até não existirem mais variáveis,
- Podemos descartar as cláusulas originais após cada iteração,
- Se no fim do método todos os literais forem puros então é devolvido SAT.

Exemplo SAT

$(\alpha \vee \beta \vee \sigma) \wedge (\beta \vee \neg\sigma) \wedge (\neg\beta \vee \gamma)$

Algoritmo de Davis Putnam

- Escolher de modo iterativo variáveis para aplicar o método de resolução até não existirem mais variáveis,
- Podemos descartar as cláusulas originais após cada iteração,
- Se no fim do método todos os literais forem puros então é devolvido SAT.

Exemplo SAT

$(\alpha \vee \beta \vee \sigma) \wedge (\beta \vee \neg\sigma) \wedge (\neg\beta \vee \gamma)$
$(\alpha \vee \gamma \vee \sigma) \wedge (\gamma \vee \neg\sigma)$

Algoritmo de Davis Putnam

- Escolher de modo iterativo variáveis para aplicar o método de resolução até não existirem mais variáveis,
- Podemos descartar as cláusulas originais após cada iteração,
- Se no fim do método todos os literais forem puros então é devolvido SAT.

Exemplo SAT

$(\alpha \vee \beta \vee \sigma) \wedge (\beta \vee \neg\sigma) \wedge (\neg\beta \vee \gamma)$
$(\alpha \vee \gamma \vee \sigma) \wedge (\gamma \vee \neg\sigma)$

Algoritmo de Davis Putnam

- Escolher de modo iterativo variáveis para aplicar o método de resolução até não existirem mais variáveis,
- Podemos descartar as cláusulas originais após cada iteração,
- Se no fim do método todos os literais forem puros então é devolvido SAT.

Exemplo SAT

$(\alpha \vee \beta \vee \sigma) \wedge (\beta \vee \neg\sigma) \wedge (\neg\beta \vee \gamma)$
$(\alpha \vee \gamma \vee \sigma) \wedge (\gamma \vee \neg\sigma)$
$(\alpha \vee \gamma)$

Algoritmo de Davis Putnam

- Escolher de modo iterativo variáveis para aplicar o método de resolução até não existirem mais variáveis,
- Podemos descartar as cláusulas originais após cada iteração,
- Se no fim do método todos os literais forem puros então é devolvido SAT.

Exemplo UNSAT

$$(\alpha \vee \beta) \wedge (\alpha \vee \neg\beta) \wedge (\neg\alpha \vee \gamma) \wedge (\neg\alpha \vee \neg\gamma)$$

Algoritmo de Davis Putnam

- Escolher de modo iterativo variáveis para aplicar o método de resolução até não existirem mais variáveis,
- Podemos descartar as cláusulas originais após cada iteração,
- Se no fim do método todos os literais forem puros então é devolvido SAT.

Exemplo UNSAT

$$(\alpha \vee \beta) \wedge (\alpha \vee \neg\beta) \wedge (\neg\alpha \vee \gamma) \wedge (\neg\alpha \vee \neg\gamma)$$

Algoritmo de Davis Putnam

- Escolher de modo iterativo variáveis para aplicar o método de resolução até não existirem mais variáveis,
- Podemos descartar as cláusulas originais após cada iteração,
- Se no fim do método todos os literais forem puros então é devolvido SAT.

Exemplo UNSAT

$(\alpha \vee \beta) \wedge (\alpha \vee \neg\beta) \wedge (\neg\alpha \vee \gamma) \wedge (\neg\alpha \vee \neg\gamma)$
$(\alpha) \wedge (\neg\alpha \vee \gamma) \wedge (\neg\alpha \vee \neg\gamma)$

Algoritmo de Davis Putnam

- Escolher de modo iterativo variáveis para aplicar o método de resolução até não existirem mais variáveis,
- Podemos descartar as cláusulas originais após cada iteração,
- Se no fim do método todos os literais forem puros então é devolvido SAT.

Exemplo UNSAT

$(\alpha \vee \beta) \wedge (\alpha \vee \neg\beta) \wedge (\neg\alpha \vee \gamma) \wedge (\neg\alpha \vee \neg\gamma)$
$(\alpha) \wedge (\neg\alpha \vee \gamma) \wedge (\neg\alpha \vee \neg\gamma)$

Algoritmo de Davis Putnam

- Escolher de modo iterativo variáveis para aplicar o método de resolução até não existirem mais variáveis,
- Podemos descartar as cláusulas originais após cada iteração,
- Se no fim do método todos os literais forem puros então é devolvido SAT.

Exemplo UNSAT

$(\alpha \vee \beta) \wedge (\alpha \vee \neg\beta) \wedge (\neg\alpha \vee \gamma) \wedge (\neg\alpha \vee \neg\gamma)$
$(\alpha) \wedge (\neg\alpha \vee \gamma) \wedge (\neg\alpha \vee \neg\gamma)$
$(\gamma) \wedge (\neg\gamma)$

Algoritmo de Davis Putnam

- Escolher de modo iterativo variáveis para aplicar o método de resolução até não existirem mais variáveis,
- Podemos descartar as cláusulas originais após cada iteração,
- Se no fim do método todos os literais forem puros então é devolvido SAT.

Exemplo UNSAT

$(\alpha \vee \beta) \wedge (\alpha \vee \neg\beta) \wedge (\neg\alpha \vee \gamma) \wedge (\neg\alpha \vee \neg\gamma)$
$(\alpha) \wedge (\neg\alpha \vee \gamma) \wedge (\neg\alpha \vee \neg\gamma)$
$(\gamma) \wedge (\neg\gamma)$

Algoritmo de Davis Putnam

- Escolher de modo iterativo variáveis para aplicar o método de resolução até não existirem mais variáveis,
- Podemos descartar as cláusulas originais após cada iteração,
- Se no fim do método todos os literais forem puros então é devolvido SAT.

Exemplo UNSAT

$(\alpha \vee \beta) \wedge (\alpha \vee \neg\beta) \wedge (\neg\alpha \vee \gamma) \wedge (\neg\alpha \vee \neg\gamma)$
$(\alpha) \wedge (\neg\alpha \vee \gamma) \wedge (\neg\alpha \vee \neg\gamma)$
$(\gamma) \wedge (\neg\gamma)$
$()$

Davis, Logemann, Loveland, 1962

- Davis, Putnam, 1960:
  - Baseado no método da resolução,
- Davis, Logemann, Loveland, 1962:
  - Baseado na pesquisa,
  - Base dos algoritmos mais recentes,
  - Aprendizagem e retrocesso não cronológico, 1996.
- Uso de técnicas SAT para outro tipo de restrições.

Regras de Dedução

Regra do Literal Unitário

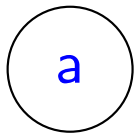
Se uma cláusula tiver todos os seus literais com o valor 0 exceptuando um então esse literal terá que ter o valor 1.

Regra do Conflito

Se uma cláusula tiver todos os seus literais com o valor 0 então a fórmula é insatisfeita naquele ramo da procura.

Algoritmo do DPLL

$$\neg a \vee b \vee c$$
$$a \vee c \vee d$$
$$a \vee c \vee \neg d$$
$$a \vee \neg c \vee d$$
$$a \vee \neg c \vee \neg d$$
$$\neg b \vee \neg c \vee d$$
$$\neg a \vee b \vee \neg c$$
$$\neg a \vee \neg b \vee c$$



Algoritmo do DPLL

$\neg a \vee b \vee c$

$a \vee c \vee d$

$a \vee c \vee \neg d$

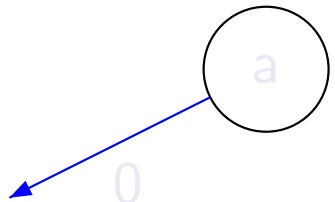
$a \vee \neg c \vee d$

$a \vee \neg c \vee \neg d$

$\neg b \vee \neg c \vee d$

$\neg a \vee b \vee \neg c$

$\neg a \vee \neg b \vee c$



Algoritmo do DPLL

$\neg a \vee b \vee c$

$a \vee c \vee d$

$a \vee c \vee \neg d$

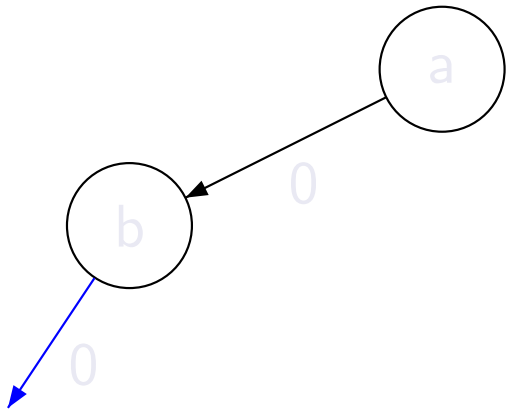
$a \vee \neg c \vee d$

$a \vee \neg c \vee \neg d$

$\neg b \vee \neg c \vee d$

$\neg a \vee b \vee \neg c$

$\neg a \vee \neg b \vee c$



Algoritmo do DPLL

$\neg a \vee b \vee c$

$a \vee c \vee d$

$a \vee c \vee \neg d$

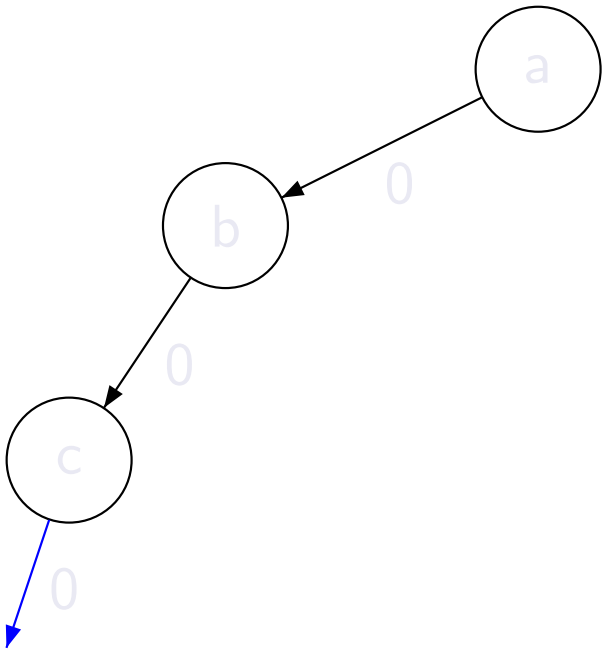
$a \vee \neg c \vee d$

$a \vee \neg c \vee \neg d$

$\neg b \vee \neg c \vee d$

$\neg a \vee b \vee \neg c$

$\neg a \vee \neg b \vee c$



Algoritmo do DPLL

$\neg a \vee b \vee c$

$a \vee c \vee d$

$a \vee c \vee \neg d$

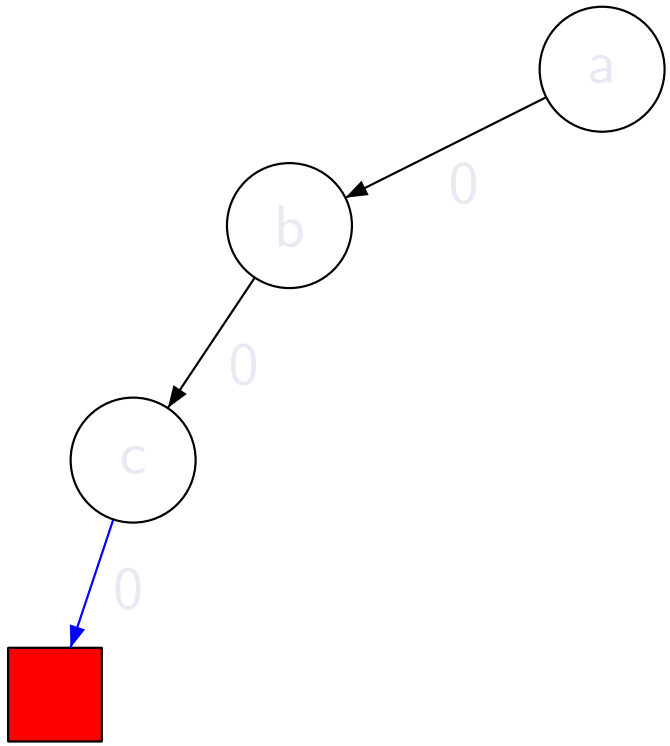
$a \vee \neg c \vee d$

$a \vee \neg c \vee \neg d$

$\neg b \vee \neg c \vee d$

$\neg a \vee b \vee \neg c$

$\neg a \vee \neg b \vee c$



Algoritmo do DPLL

$\neg a \vee b \vee c$

$a \vee c \vee d$

$a \vee c \vee \neg d$

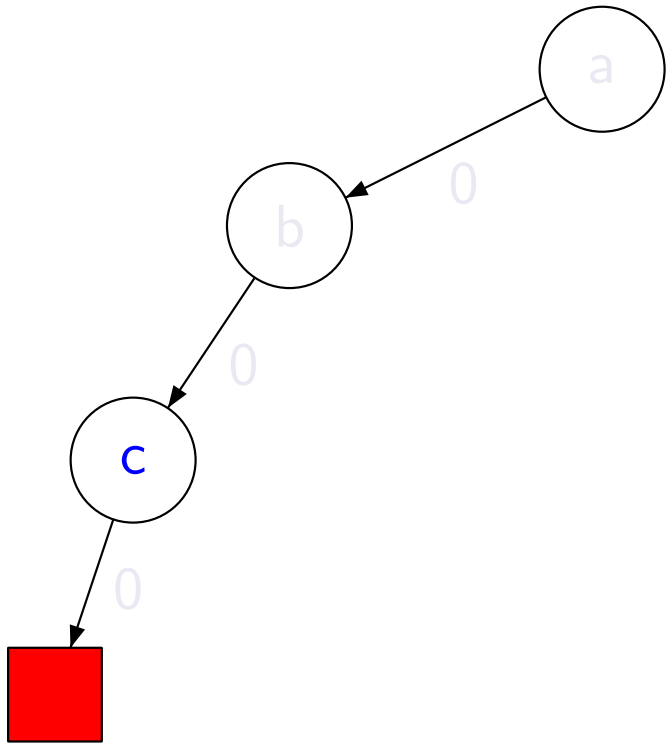
$a \vee \neg c \vee d$

$a \vee \neg c \vee \neg d$

$\neg b \vee \neg c \vee d$

$\neg a \vee b \vee \neg c$

$\neg a \vee \neg b \vee c$



Algoritmo do DPLL

$\neg a \vee b \vee c$

$a \vee c \vee d$

$a \vee c \vee \neg d$

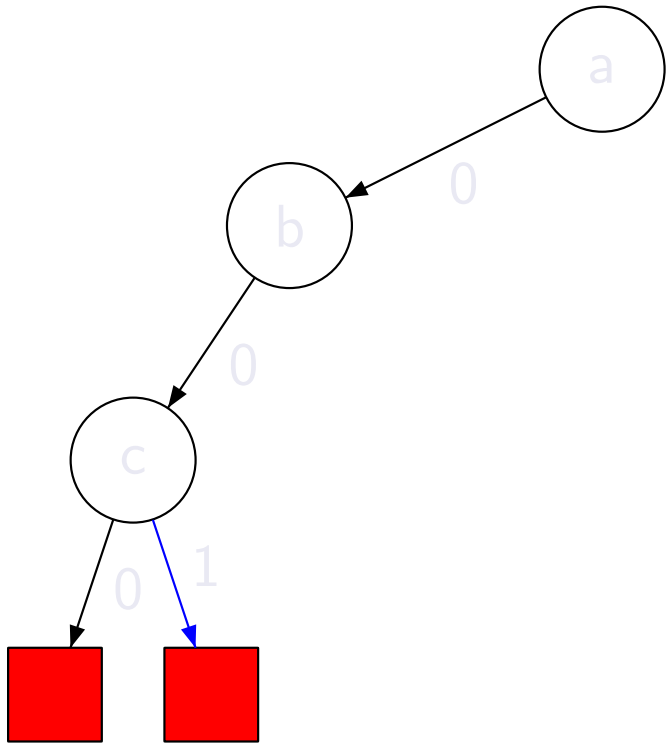
$a \vee \neg c \vee d$

$a \vee \neg c \vee \neg d$

$\neg b \vee \neg c \vee d$

$\neg a \vee b \vee \neg c$

$\neg a \vee \neg b \vee c$



Algoritmo do DPLL

$\neg a \vee b \vee c$

$a \vee c \vee d$

$a \vee c \vee \neg d$

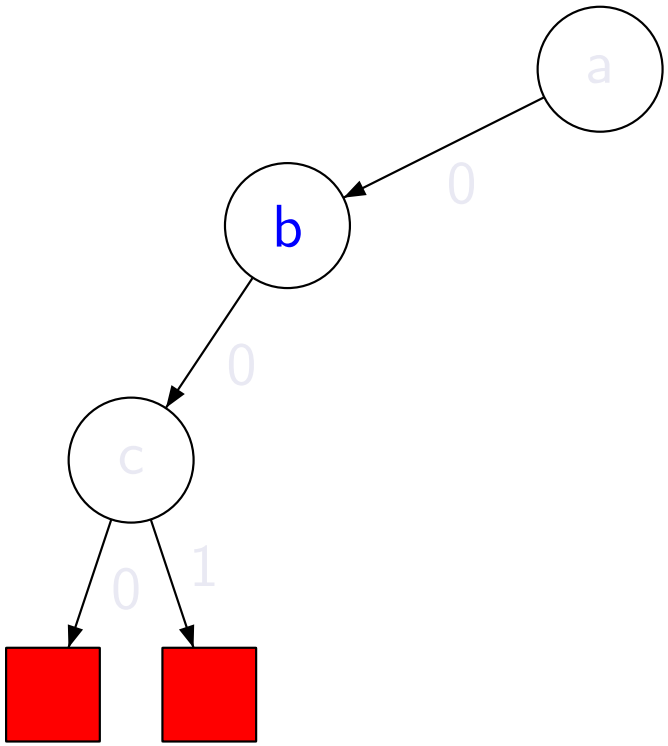
$a \vee \neg c \vee d$

$a \vee \neg c \vee \neg d$

$\neg b \vee \neg c \vee d$

$\neg a \vee b \vee \neg c$

$\neg a \vee \neg b \vee c$



Algoritmo do DPLL

$\neg a \vee b \vee c$

$a \vee c \vee d$

$a \vee c \vee \neg d$

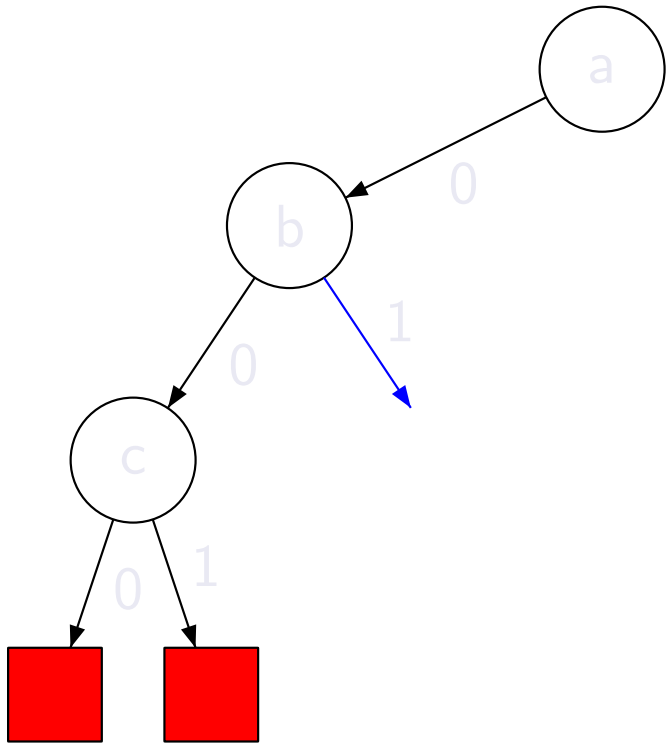
$a \vee \neg c \vee d$

$a \vee \neg c \vee \neg d$

$\neg b \vee \neg c \vee d$

$\neg a \vee b \vee \neg c$

$\neg a \vee \neg b \vee c$



Algoritmo do DPLL

$\neg a \vee b \vee c$

$a \vee c \vee d$

$a \vee c \vee \neg d$

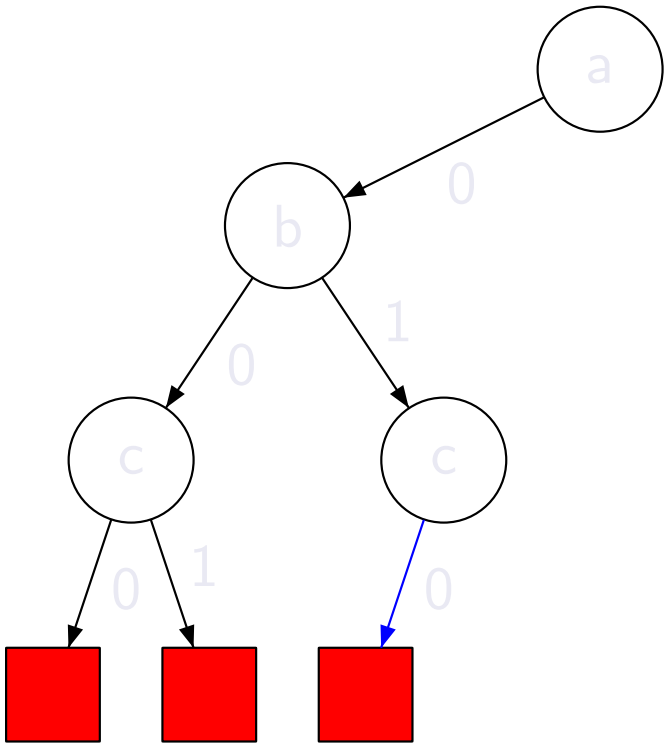
$a \vee \neg c \vee d$

$a \vee \neg c \vee \neg d$

$\neg b \vee \neg c \vee d$

$\neg a \vee b \vee \neg c$

$\neg a \vee \neg b \vee c$



Algoritmo do DPLL

$\neg a \vee b \vee c$

$a \vee c \vee d$

$a \vee c \vee \neg d$

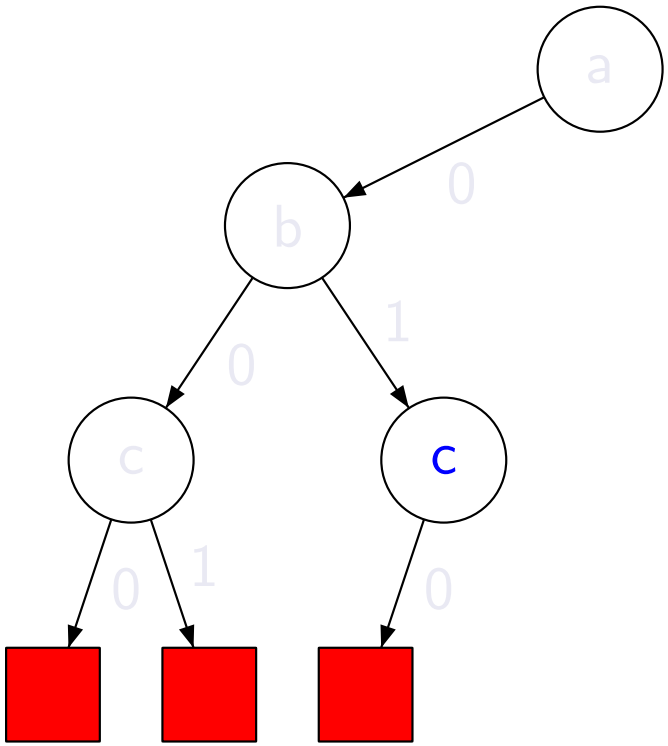
$a \vee \neg c \vee d$

$a \vee \neg c \vee \neg d$

$\neg b \vee \neg c \vee d$

$\neg a \vee b \vee \neg c$

$\neg a \vee \neg b \vee c$



Algoritmo do DPLL

$\neg a \vee b \vee c$

$a \vee c \vee d$

$a \vee c \vee \neg d$

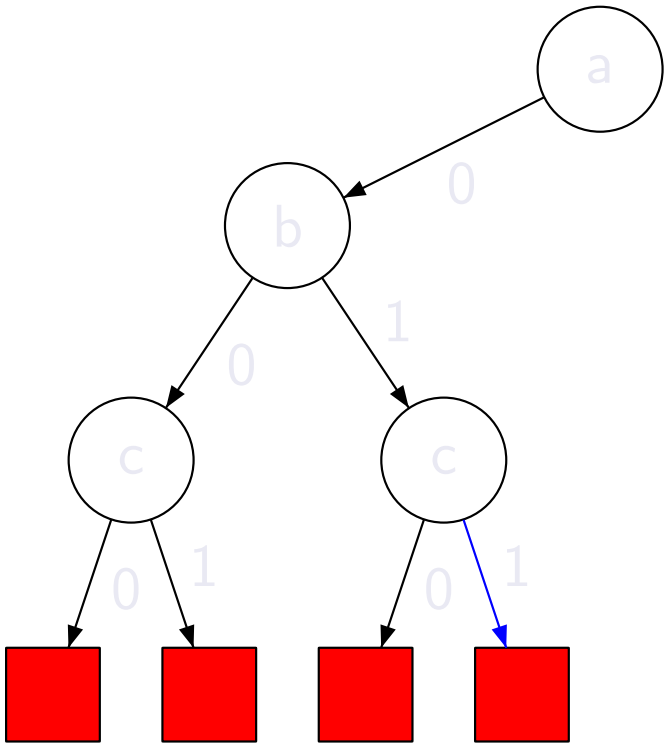
$a \vee \neg c \vee d$

$a \vee \neg c \vee \neg d$

$\neg b \vee \neg c \vee d$

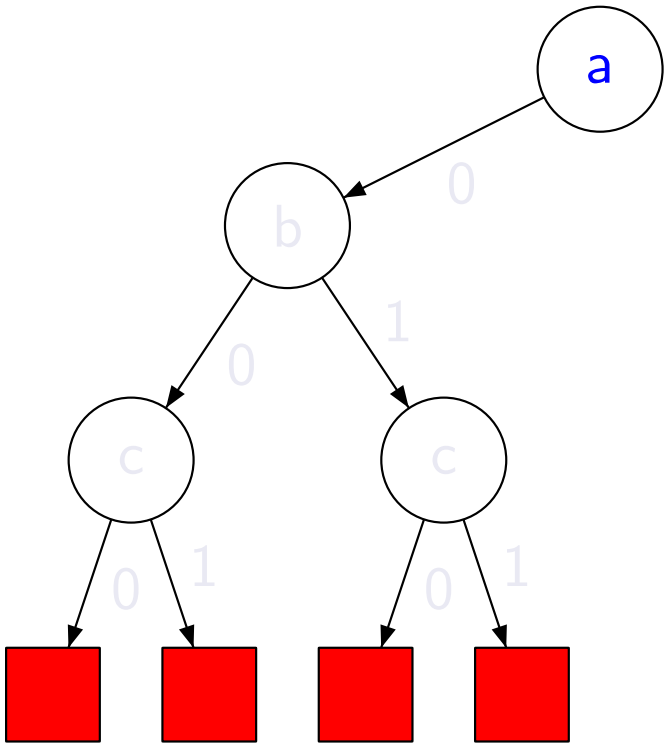
$\neg a \vee b \vee \neg c$

$\neg a \vee \neg b \vee c$



Algoritmo do DPLL

$$\neg a \vee b \vee c$$
$$a \vee c \vee d$$
$$a \vee c \vee \neg d$$
$$a \vee \neg c \vee d$$
$$a \vee \neg c \vee \neg d$$
$$\neg b \vee \neg c \vee d$$
$$\neg a \vee b \vee \neg c$$
$$\neg a \vee \neg b \vee c$$



Algoritmo do DPLL

$\neg a \vee b \vee c$

$a \vee c \vee d$

$a \vee c \vee \neg d$

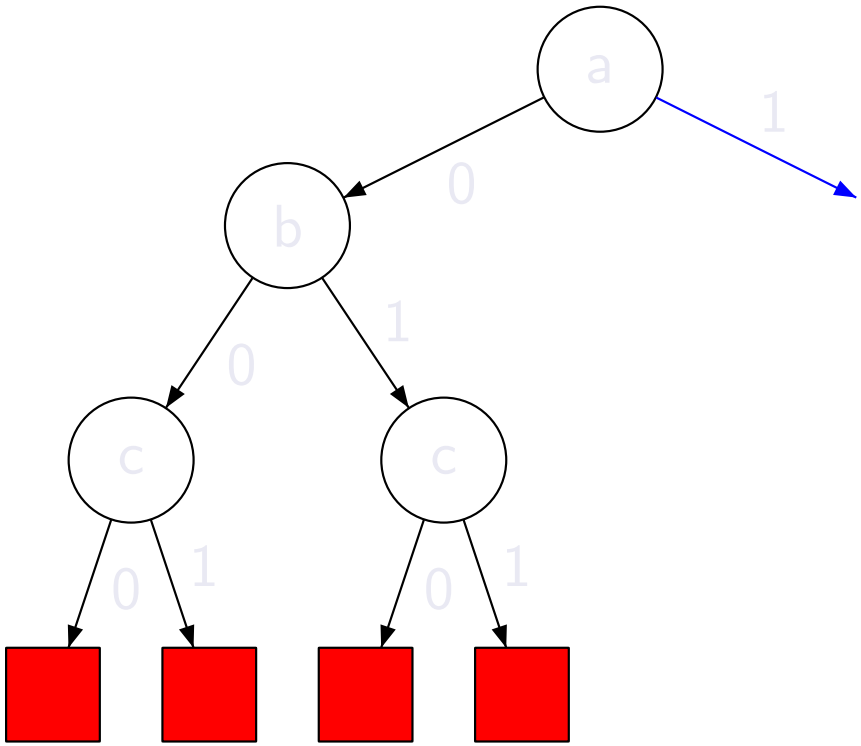
$a \vee \neg c \vee d$

$a \vee \neg c \vee \neg d$

$\neg b \vee \neg c \vee d$

$\neg a \vee b \vee \neg c$

$\neg a \vee \neg b \vee c$



Algoritmo do DPLL

$\neg a \vee b \vee c$

$a \vee c \vee d$

$a \vee c \vee \neg d$

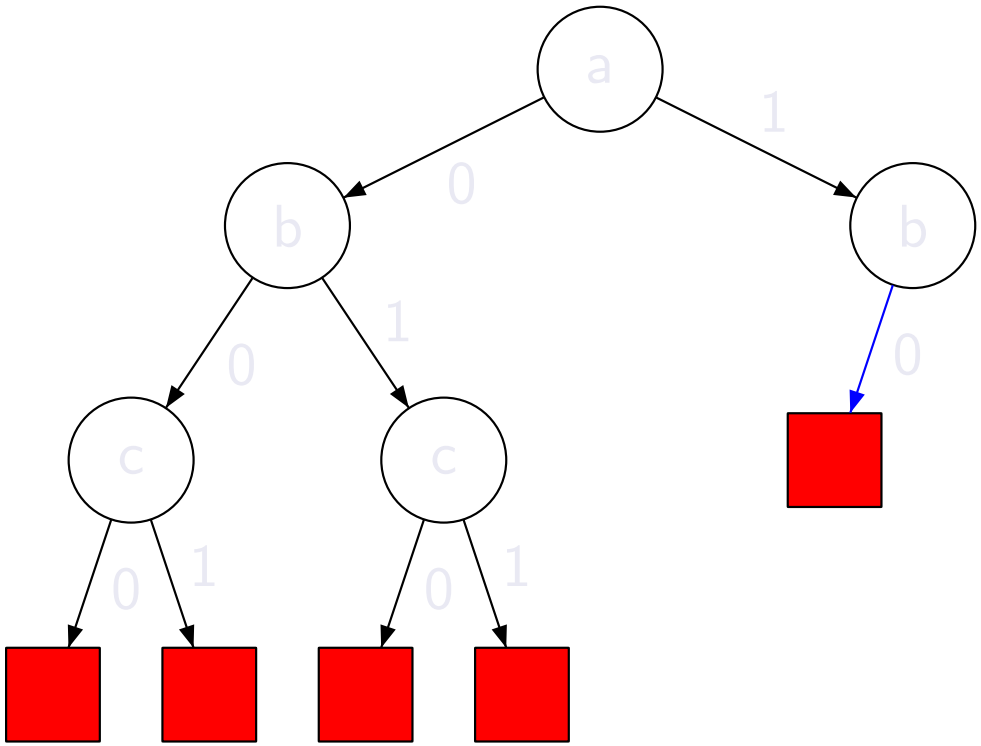
$a \vee \neg c \vee d$

$a \vee \neg c \vee \neg d$

$\neg b \vee \neg c \vee d$

$\neg a \vee b \vee \neg c$

$\neg a \vee \neg b \vee c$



Algoritmo do DPLL

$\neg a \vee b \vee c$

$a \vee c \vee d$

$a \vee c \vee \neg d$

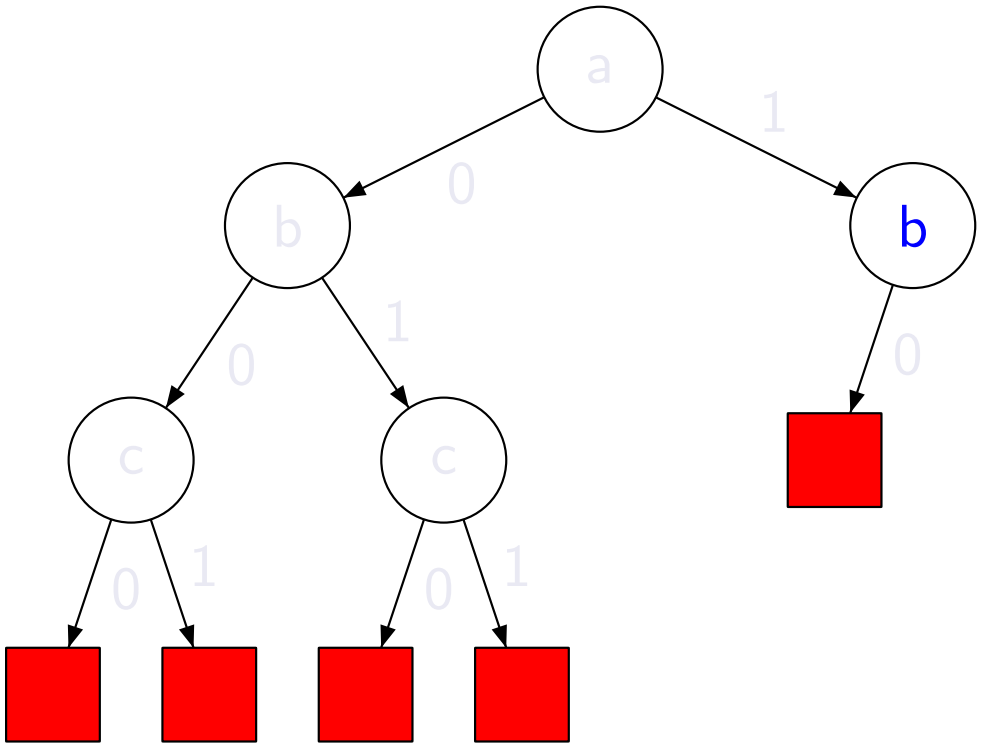
$a \vee \neg c \vee d$

$a \vee \neg c \vee \neg d$

$\neg b \vee \neg c \vee d$

$\neg a \vee b \vee \neg c$

$\neg a \vee \neg b \vee c$



Algoritmo do DPLL

$\neg a \vee b \vee c$

$a \vee c \vee d$

$a \vee c \vee \neg d$

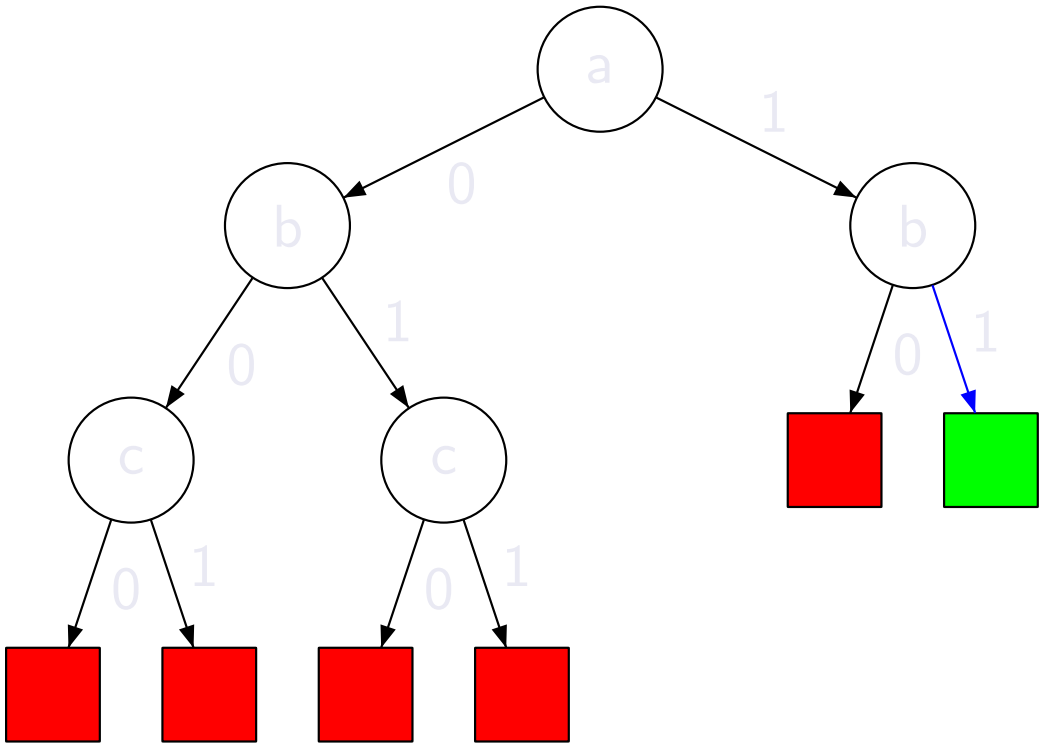
$a \vee \neg c \vee d$

$a \vee \neg c \vee \neg d$

$\neg b \vee \neg c \vee d$

$\neg a \vee b \vee \neg c$

$\neg a \vee \neg b \vee c$



Aprendizagem por Confiltos e Retrocesso Não Cronológico

$\neg a \vee b \vee c$

$a \vee c \vee d$

$a \vee c \vee \neg d$

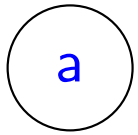
$a \vee \neg c \vee d$

$a \vee \neg c \vee \neg d$

$\neg b \vee \neg c \vee d$

$\neg a \vee b \vee \neg c$

$\neg a \vee \neg b \vee c$



Aprendizagem por Confiltos e Retrocesso Não Cronológico

$\neg a \vee b \vee c$

$a \vee c \vee d$

$a \vee c \vee \neg d$

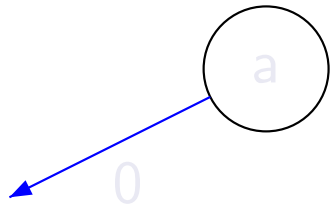
$a \vee \neg c \vee d$

$a \vee \neg c \vee \neg d$

$\neg b \vee \neg c \vee d$

$\neg a \vee b \vee \neg c$

$\neg a \vee \neg b \vee c$



Aprendizagem por Confiltos e Retrocesso Não Cronológico

$\neg a \vee b \vee c$

$a \vee c \vee d$

$a \vee c \vee \neg d$

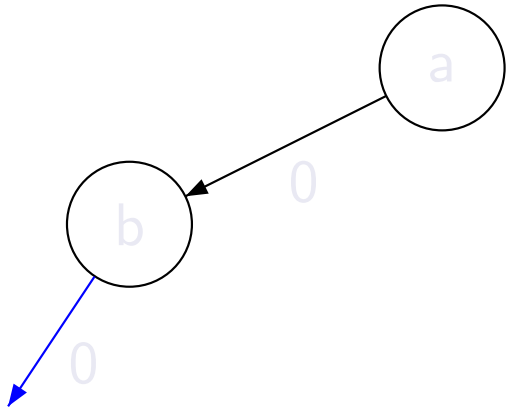
$a \vee \neg c \vee d$

$a \vee \neg c \vee \neg d$

$\neg b \vee \neg c \vee d$

$\neg a \vee b \vee \neg c$

$\neg a \vee \neg b \vee c$



Aprendizagem por Confiltos e Retrocesso Não Cronológico

$\neg a \vee b \vee c$

$a \vee c \vee d$

$a \vee c \vee \neg d$

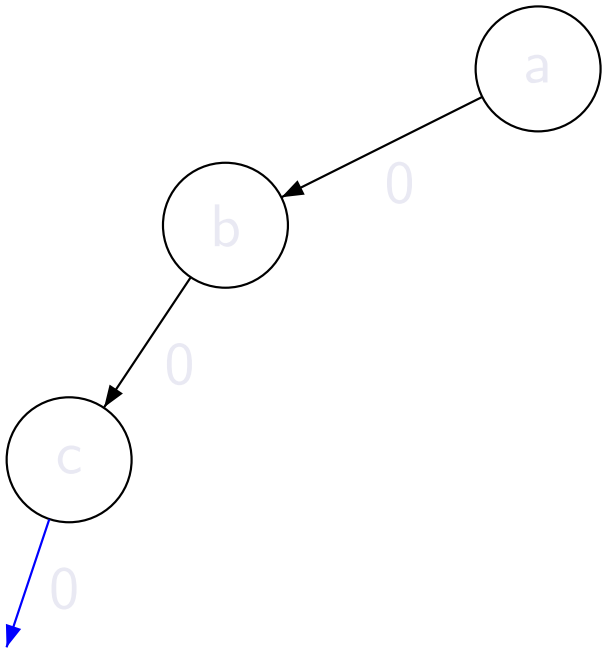
$a \vee \neg c \vee d$

$a \vee \neg c \vee \neg d$

$\neg b \vee \neg c \vee d$

$\neg a \vee b \vee \neg c$

$\neg a \vee \neg b \vee c$



Aprendizagem por Confiltos e Retrocesso Não Cronológico

$\neg a \vee b \vee c$

$a \vee c \vee d$

$a \vee c \vee \neg d$

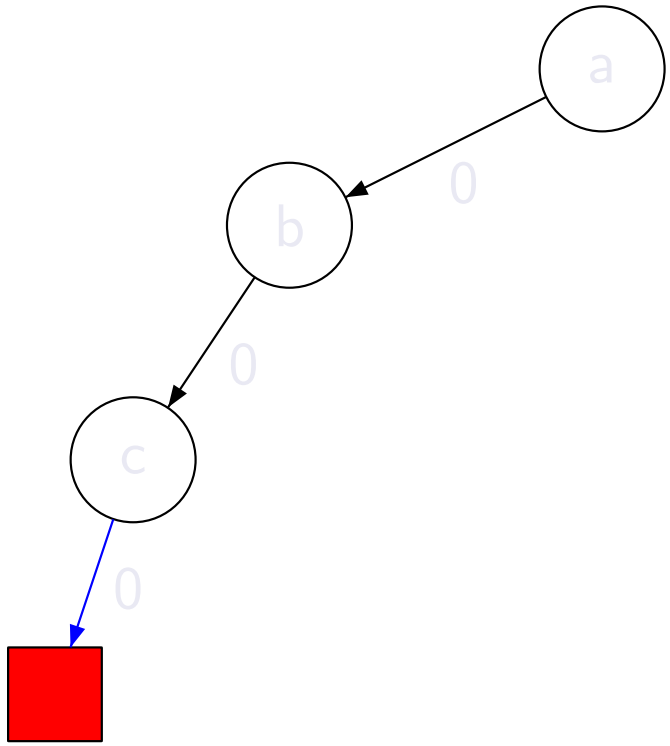
$a \vee \neg c \vee d$

$a \vee \neg c \vee \neg d$

$\neg b \vee \neg c \vee d$

$\neg a \vee b \vee \neg c$

$\neg a \vee \neg b \vee c$



Aprendizagem por Confiltos e Retrocesso Não Cronológico

$\neg a \vee b \vee c$

$a \vee c \vee d$

$a \vee c \vee \neg d$

$a \vee \neg c \vee d$

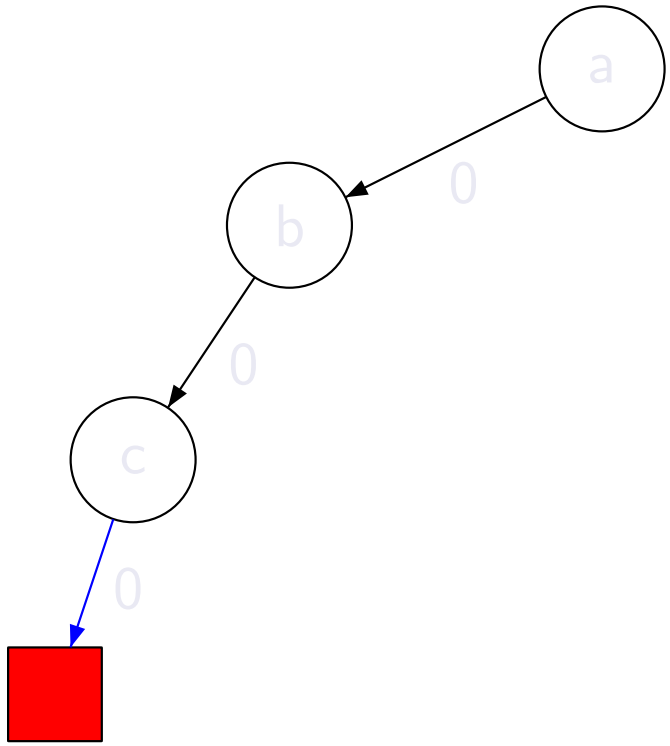
$a \vee \neg c \vee \neg d$

$\neg b \vee \neg c \vee d$

$\neg a \vee b \vee \neg c$

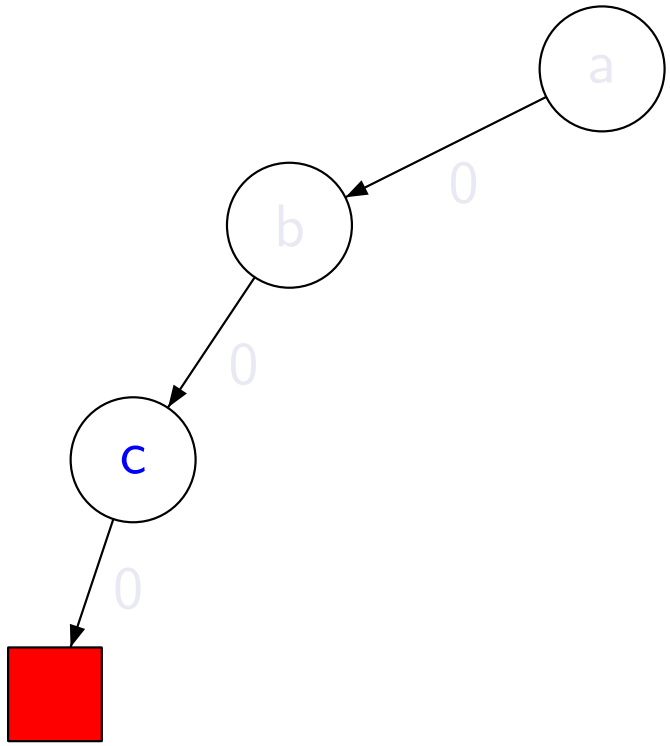
$\neg a \vee \neg b \vee c$

$a \vee c$



Aprendizagem por Confiltos e Retrocesso Não Cronológico

$\neg a \vee b \vee c$
$a \vee c \vee d$
$a \vee c \vee \neg d$
$a \vee \neg c \vee d$
$a \vee \neg c \vee \neg d$
$\neg b \vee \neg c \vee d$
$\neg a \vee b \vee \neg c$
$\neg a \vee \neg b \vee c$
$a \vee c$



Aprendizagem por Confiltos e Retrocesso Não Cronológico

$\neg a \vee b \vee c$

$a \vee c \vee d$

$a \vee c \vee \neg d$

$a \vee \neg c \vee d$

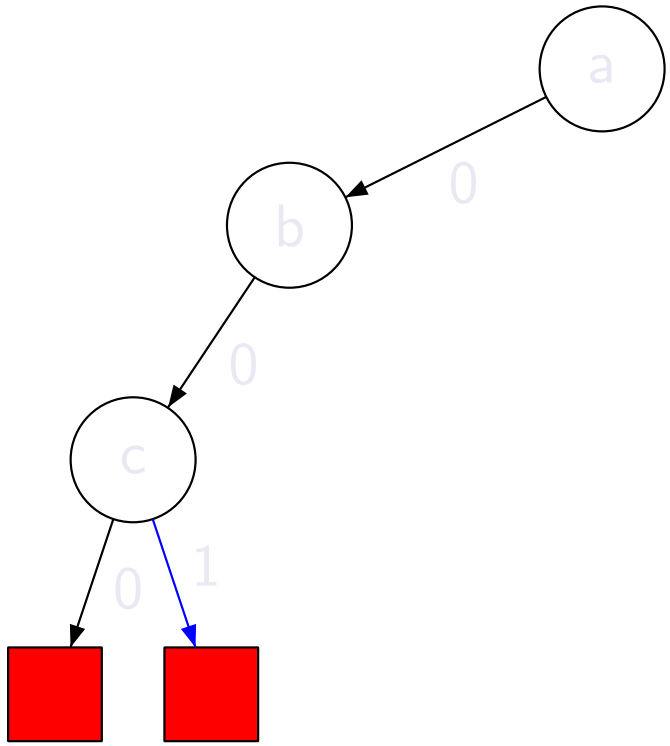
$a \vee \neg c \vee \neg d$

$\neg b \vee \neg c \vee d$

$\neg a \vee b \vee \neg c$

$\neg a \vee \neg b \vee c$

$a \vee c$



Aprendizagem por Confiltos e Retrocesso Não Cronológico

$\neg a \vee b \vee c$

$a \vee c \vee d$

$a \vee c \vee \neg d$

$a \vee \neg c \vee d$

$a \vee \neg c \vee \neg d$

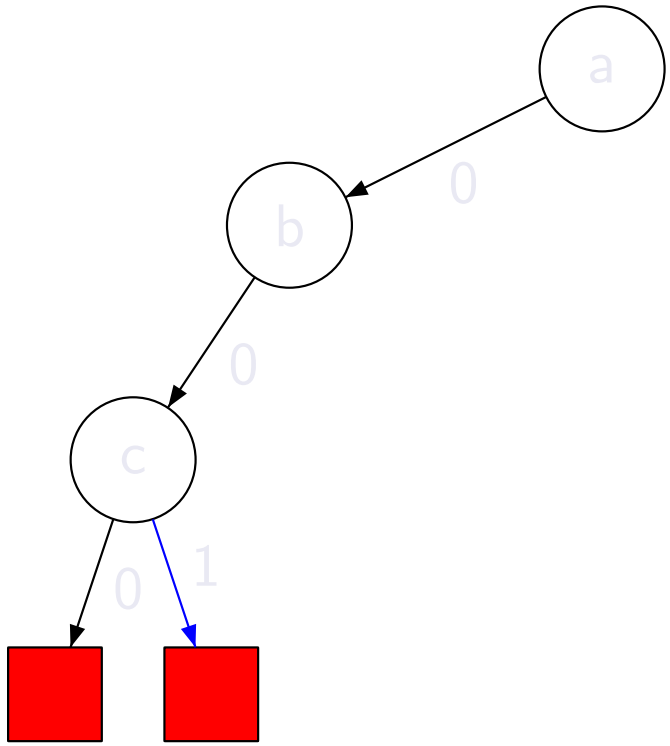
$\neg b \vee \neg c \vee d$

$\neg a \vee b \vee \neg c$

$\neg a \vee \neg b \vee c$

$a \vee c$

$a \vee \neg c$



Aprendizagem por Confiltos e Retrocesso Não Cronológico

$\neg a \vee b \vee c$

$a \vee c \vee d$

$a \vee c \vee \neg d$

$a \vee \neg c \vee d$

$a \vee \neg c \vee \neg d$

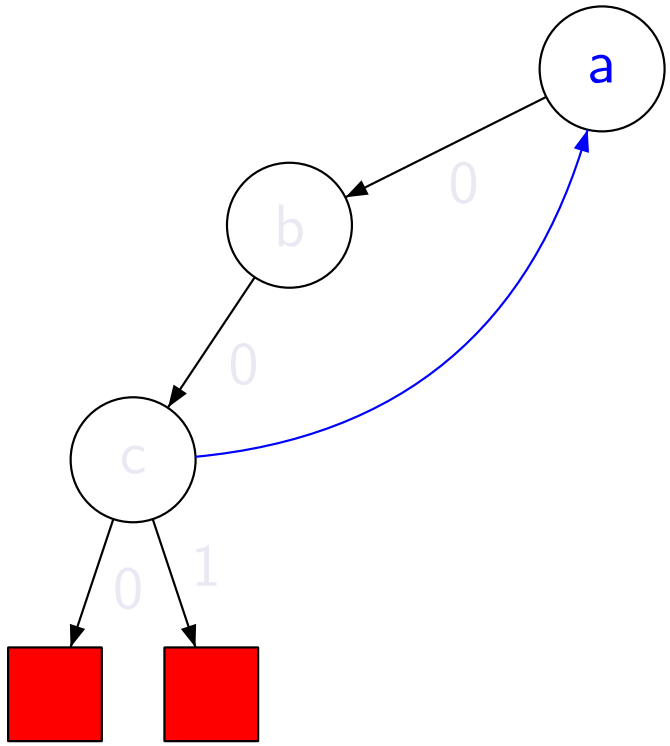
$\neg b \vee \neg c \vee d$

$\neg a \vee b \vee \neg c$

$\neg a \vee \neg b \vee c$

$a \vee c$

$a \vee \neg c$



Aprendizagem por Confiltos e Retrocesso Não Cronológico

$\neg a \vee b \vee c$

$a \vee c \vee d$

$a \vee c \vee \neg d$

$a \vee \neg c \vee d$

$a \vee \neg c \vee \neg d$

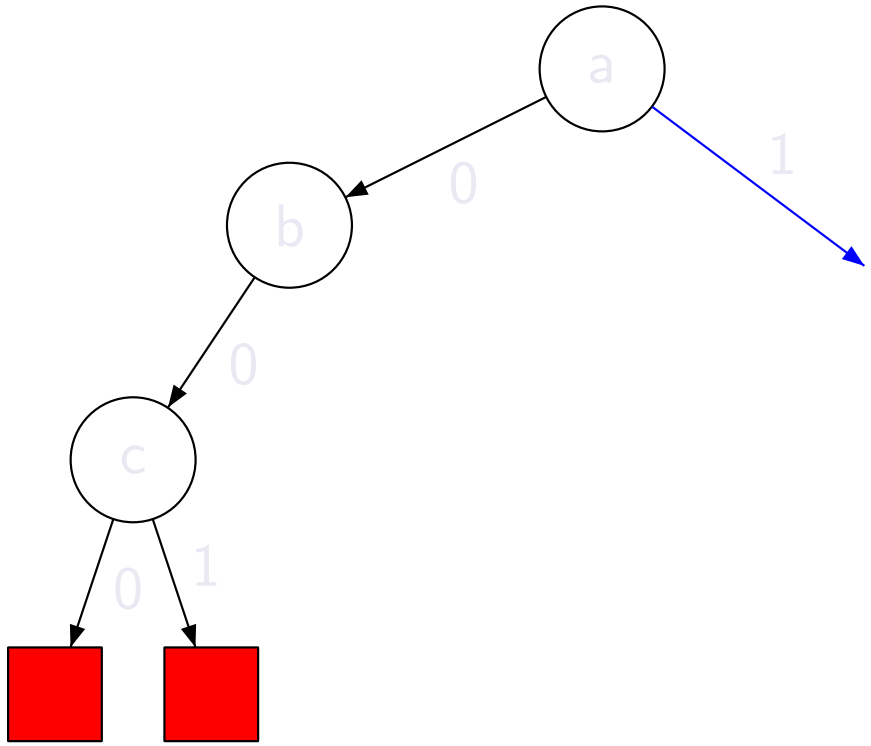
$\neg b \vee \neg c \vee d$

$\neg a \vee b \vee \neg c$

$\neg a \vee \neg b \vee c$

$a \vee c$

$a \vee \neg c$



Aprendizagem por Confiltos e Retrocesso Não Cronológico

$\neg a \vee b \vee c$

$a \vee c \vee d$

$a \vee c \vee \neg d$

$a \vee \neg c \vee d$

$a \vee \neg c \vee \neg d$

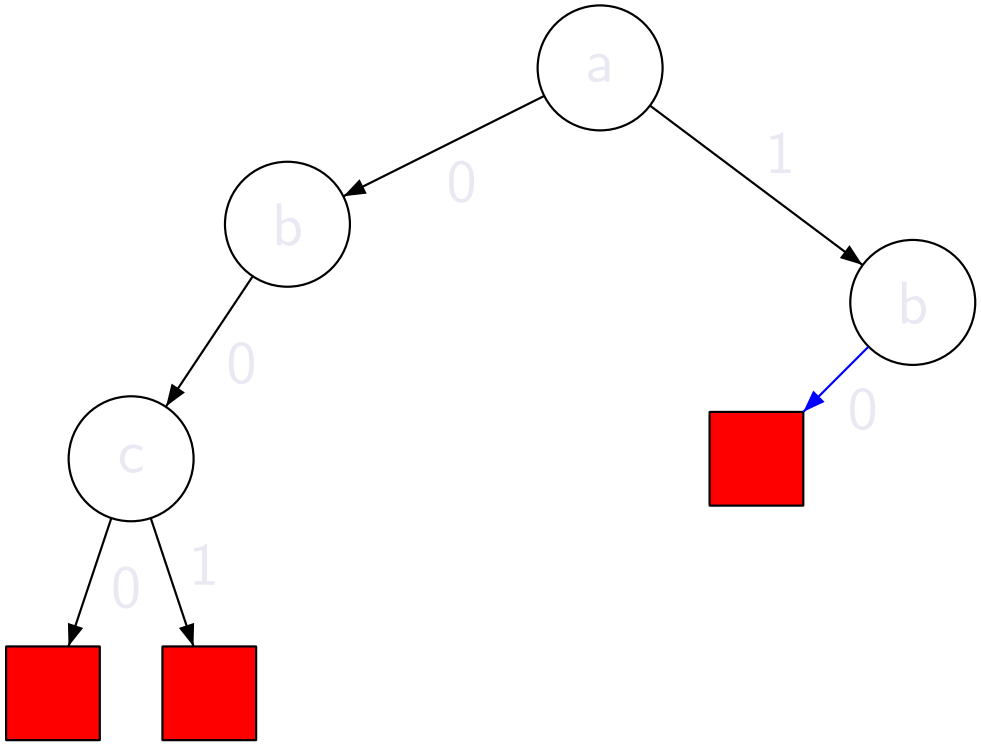
$\neg b \vee \neg c \vee d$

$\neg a \vee b \vee \neg c$

$\neg a \vee \neg b \vee c$

$a \vee c$

$a \vee \neg c$



Aprendizagem por Confiltos e Retrocesso Não Cronológico

$\neg a \vee b \vee c$

$a \vee c \vee d$

$a \vee c \vee \neg d$

$a \vee \neg c \vee d$

$a \vee \neg c \vee \neg d$

$\neg b \vee \neg c \vee d$

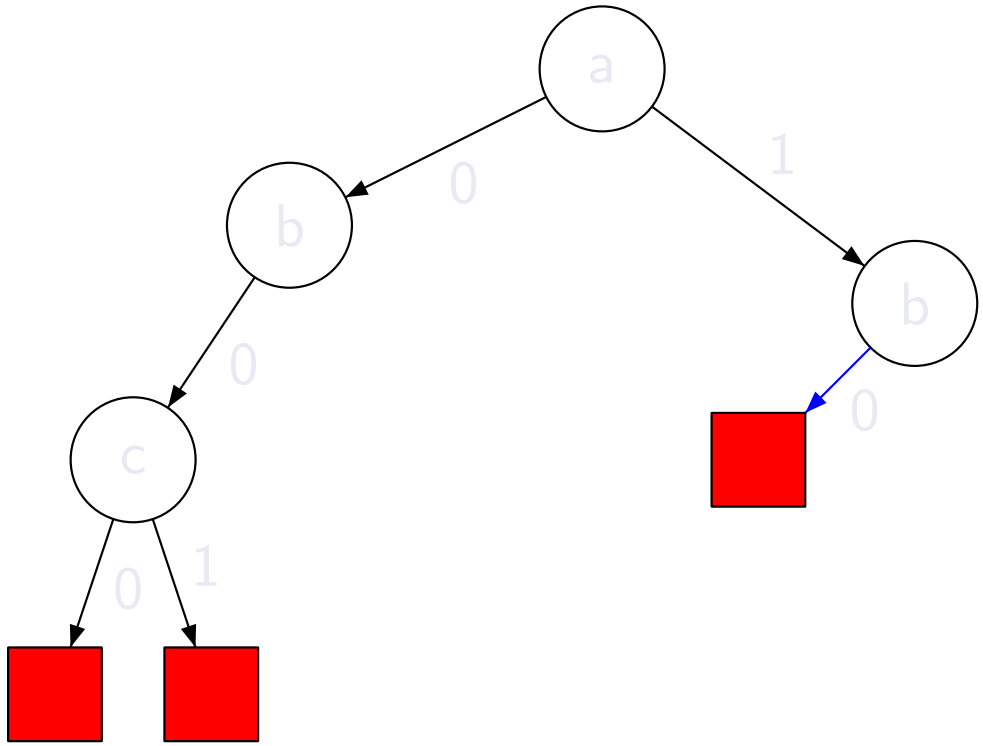
$\neg a \vee b \vee \neg c$

$\neg a \vee \neg b \vee c$

$a \vee c$

$a \vee \neg c$

$\neg a \vee b$



Aprendizagem por Confiltos e Retrocesso Não Cronológico

$\neg a \vee b \vee c$

$a \vee c \vee d$

$a \vee c \vee \neg d$

$a \vee \neg c \vee d$

$a \vee \neg c \vee \neg d$

$\neg b \vee \neg c \vee d$

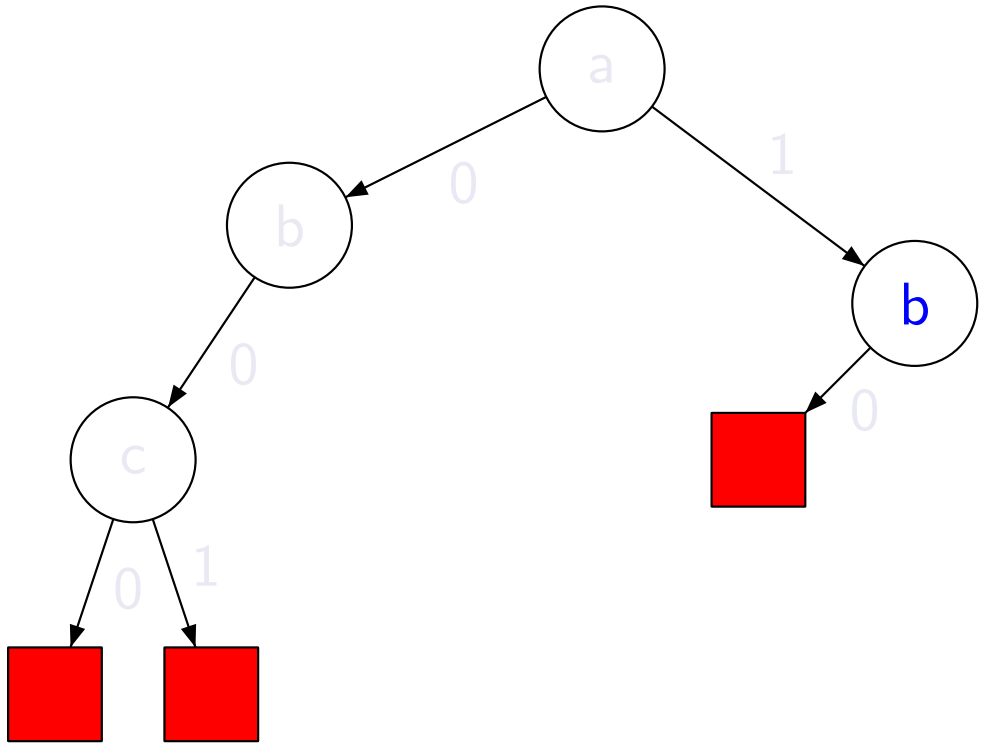
$\neg a \vee b \vee \neg c$

$\neg a \vee \neg b \vee c$

$a \vee c$

$a \vee \neg c$

$\neg a \vee b$



Aprendizagem por Confiltos e Retrocesso Não Cronológico

$\neg a \vee b \vee c$

$a \vee c \vee d$

$a \vee c \vee \neg d$

$a \vee \neg c \vee d$

$a \vee \neg c \vee \neg d$

$\neg b \vee \neg c \vee d$

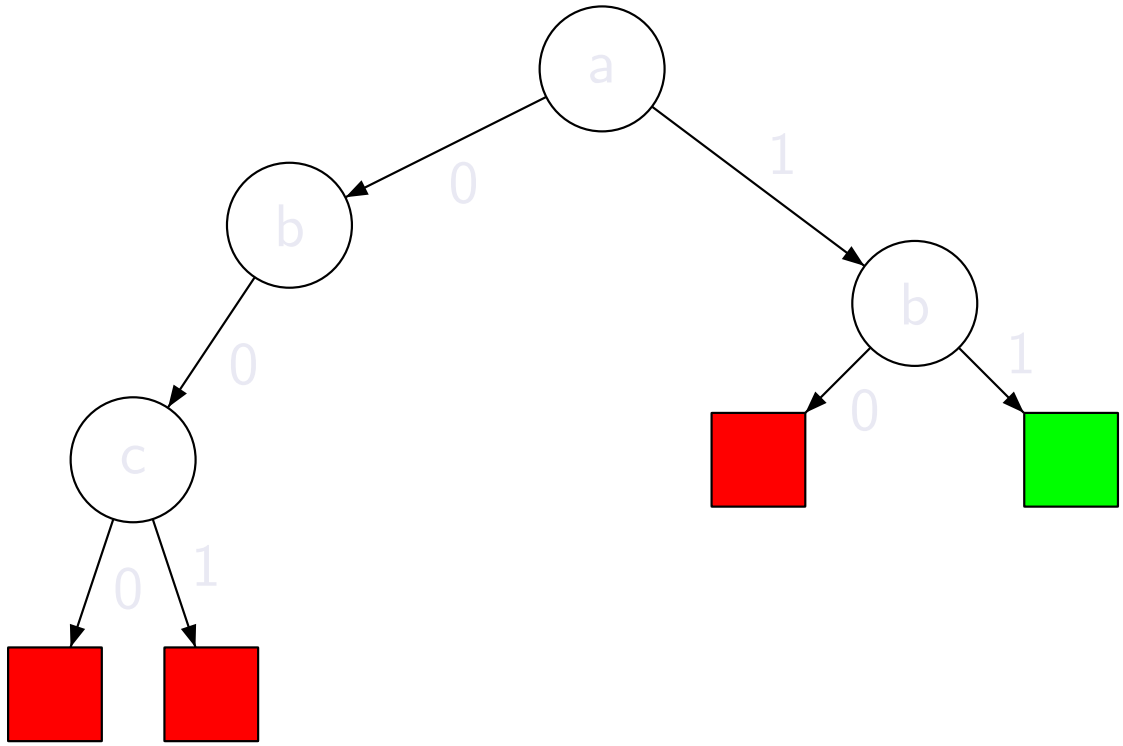
$\neg a \vee b \vee \neg c$

$\neg a \vee \neg b \vee c$

$a \vee c$

$a \vee \neg c$

$\neg a \vee b$



Métodos Estocásticos, 1992

- Davis, Putnam, 1960:
  - Baseado no método da resolução,
- Davis, Logemann, Loveland, 1962:
  - Baseado na pesquisa,
  - Base dos algoritmos mais recentes,
  - Aprendizagem e retrocesso não cronológico, 1996.
- Uso de técnicas SAT para outro tipo de restrições.

Uso de técnicas SAT para outro tipo de restrições

- Limitações de SAT:
  - As variáveis estão no domínio booleano,
  - As restrições são expressas como cláusulas.
- Vantagens de SAT:
  - Algoritmos eficientes com base no DPLL,
  - Propagação de variáveis é feita de forma rápida,
  - Aprendizagem e retrocesso não cronológico.
- Será que podemos remover algumas das limitações mantendo a eficácia das técnicas SAT conhecidas?
  - Domínio de dimensão variável,
  - Restrições Pseudo-Booleanas.

Restrições Pseudo-Booleanas

Restrições Pseudo-Booleanas

$c_1 x_1 \ldots c_n x_n \tilde{} g$

$c_i, g \in Z$; $\tilde{} \in \{=, \leq, \geq\}$; $x_i \in$ *Literais*

Exemplos

$3x_1 + x_2 = 2$

$4x_1 + 5(\neg x_2) + x_3 \leq 3$

Restrições Pseudo-Booleanas

- Cláusulas podem ser escritas como restrições Pseudo-Booleanas:

$$(\alpha \vee \beta) \rightarrow (\alpha + \beta \geq 1)$$

- O tipo de coeficientes e o tipo de restrições fazem com que a manipulação seja mais fácil.

Modelação

- Algoritmos SAT eficientes, no entanto, a eficiência dos algoritmos é apenas uma parte da resolução do problema.
- Importância na codificação do Problema:
  - Diferentes codificações podem ter grande impacto na resolução do problema.
- Muitos problemas têm um número elevado de soluções equivalentes:
  - Eliminação de soluções equivalentes.

Introdução

- Um jogo de lógica e racíocinio com regras simples,
- Sudoku ou Su Doku, significa em japonês, 'Número Único',
- Apareceu inicialmente em 1984 no Japão, ganhando popularidade neste último ano.

Regras

- Dada uma matriz 9×9 com alguns locais preenchidos:
  - Em cada linha aparecem os números 1 a 9,
  - Em cada coluna aparecem os números 1 a 9,
  - Existem 9 blocos 3×3. Em cada bloco aparecem os números 1 a 9.

Regras

Exemplo

							1	
4								
	2							
				5		4		7
		8				3		
		1		9				
3			4			2		
	5		1					
			8		6			

Regras

Exemplo

6	9	3	7	8	4	5	1	2
4	8	7	5	1	2	9	3	6
1	2	5	9	6	3	8	7	4
9	3	2	6	5	1	4	8	7
5	6	8	2	4	7	3	9	1
7	4	1	3	9	8	6	2	5
3	1	9	4	7	5	2	6	8
8	5	6	1	2	9	7	4	3
2	7	4	8	3	6	1	5	9

Sudoku Minimal

- Qual é o número mínimo de casas preenchidas tal que o Sudoku apenas tenha uma solução?
  - O número mínimo conhecido até ao momento é 17,
  - São conhecidos 36628 Sudokus-17 não equivalentes:
    - http://www.csse.uwa.edu.au/~gordon/sudokumin.php
  - 6,670,903,752,021,072,936,960 soluções para o Sudoku,
  - 5,472,730,538 soluções, considerando as simetrias.

Sudoku Minimal

Sudokus Equivalentes

- Permutação dos 9 números,
- Troca de colunas e linhas,
- Permutação de colunas num bloco,
- Permutação de linhas num bloco,
- Permutação de blocos relativamente às linhas,
- Permutação de blocos relativamente às colunas.

Complexidade

- A resolução do Sudoku em puzzles $n^2 \times n^2$ com $n \times n$ blocos é um problema NP-Completo.
- Sudoku pode ser reduzido a um problema de coloração de grafos:
  - Grafo com 81 vértices,
  - Cada vértice é um par (x,y),
  - As arestas correspondem aos pares que estão na mesma coluna, linha ou bloco.

Abordagem SAT

- 9 variáveis para cada local da matriz 9×9
  - 729 variáveis
- As variáveis são da forma S_{xyz}:
  - $S_{xyz} = 1$ quando na linha x e na coluna y está o valor z.
- Codificações:
  - Mínima,
  - Estendida.

Codificação Mínima

- Existe pelo menos um número em cada entrada:

$$\bigwedge_{x=1}^{9} \bigwedge_{y=1}^{9} \bigvee_{z=1}^{9} S_{xyz}$$

- Cada número aparece no máximo uma vez em cada linha:

$$\bigwedge_{y=1}^{9} \bigwedge_{z=1}^{9} \bigwedge_{x=1}^{8} \bigwedge_{i=x+1}^{9} (\neg S_{xyz} \vee \neg S_{iyz})$$

Codificação Mínima

- Cada número aparece no máximo uma vez em cada coluna:

$$\bigwedge_{x=1}^{9} \bigwedge_{z=1}^{9} \bigwedge_{y=1}^{8} \bigwedge_{i=y+1}^{9} (\neg S_{xyz} \vee \neg S_{xiz})$$

- Cada número aparece uma vez em cada bloco 3×3:

$$\bigwedge_{z=1}^{9} \bigwedge_{i=0}^{2} \bigwedge_{j=0}^{2} \bigwedge_{x=1}^{3} \bigwedge_{y=1}^{3} \bigwedge_{k=y+1}^{3} (\neg S_{(3i+x)(3j+y)z} \vee \neg S_{(3i+x)(3j+k)z})$$

$$\bigwedge_{z=1}^{9} \bigwedge_{i=0}^{2} \bigwedge_{j=0}^{2} \bigwedge_{x=1}^{3} \bigwedge_{y=1}^{3} \bigwedge_{k=y+1}^{3} \bigwedge_{l=1}^{3} (\neg S_{(3i+x)(3j+y)z} \vee \neg S_{(3i+k)(3j+l)z})$$

Codificação Estendida

- Existe no máximo um número em cada entrada:

$$\bigwedge_{x=1}^{9} \bigwedge_{y=1}^{9} \bigwedge_{z=1}^{8} \bigwedge_{i=z+1}^{9} (\neg S_{xyz} \vee \neg S_{xyi})$$

- Cada número aparece pelo menos uma vez em cada linha:

$$\bigwedge_{y=1}^{9} \bigwedge_{z=1}^{9} \bigvee_{x=1}^{9} S_{xyz}$$

Codificação Estendida

- Cada número aparece pelo menos uma vez em cada coluna:

$$\bigwedge_{x=1}^{9} \bigwedge_{z=1}^{9} \bigvee_{y=1}^{9} S_{xyz}$$

- Cada número aparece pelo menos uma vez em cada bloco 3×3:

$$\bigvee_{z=1}^{9} \bigwedge_{i=0}^{2} \bigwedge_{j=0}^{2} \bigwedge_{x=1}^{3} \bigwedge_{y=1}^{3} S_{(3i+x)(3j+y)z}$$

Codificação Mínima vs Estendida

- Tamanho de ambas na ordem de $O(n^4)$
- Mínima:
  - 8829 Cláusulas.
- Estendida:
  - 11998 Cláusulas.

Técnicas de Inferência

Propagação Unitária (up)

- Cláusula Unitária $w_i = (l_j)$,
- Todas as Cláusulas que contenham o literal l_j são satisfeitas,
- O literal $\neg l_j$ é removido de todas as cláusulas,
- Podemos aplicar esta técnica de modo iterativo.

Regra do Literal Falhado (flr)

- Forçamos uma atribuição a uma variável $x = v(x)$,
- Por Propagação Unitária chegamos a um conflito,
- Concluímos que $x = 1 - v(x)$.

Técnicas de Inferência

Regra Binária do Literal Falhado (binflr)

- $x_i = v(x_i)$ e $x_j = v(x_j)$,
- $(x_i = 1 - v(x_i)) \vee (x_j = 1 - v(x_j))$.

Resolução Hiper-Binária (hypre)

- $(\neg l_1 \vee x_i) \wedge (\neg l_2 \vee x_i) \wedge \ldots \wedge (\neg l_k \vee x_i) \wedge (l_1 \vee l_2 \vee \ldots \vee l_k \vee x_j)$,
- $(x_i \vee x_j)$.

Resultados Experimentais

Resultados Experimentais

	up	up+flr	up+hypre	up+binflr
Mínima	0%	1%	25%	69%
Estendida	47%	100%	100%	100%

- Propagação Unitária (up)
- Regra do Literal Falhado (flr)
- Regra Binária do Literal Falhado (binflr)
- Resolução Hiper-Binária (hypre)

Social Golfer

- O clube de golfe tem 32 membros,
- Cada membro joga uma vez por semana,
- Os membros jogam sempre em grupos de 4,
- Nenhum jogador joga no mesmo grupo do que outro jogador duas ou mais vezes.
- Generalização do Problema (w-p-g):
  - w: semanas,
  - p: número de jogadores por grupo,
  - g: número de grupos.

Social Golfer, $3 - 2 - 2$

Semana	Grupo 1	Grupo 2
1	1 2	3 4
2	1 3	2 4
3	1 4	2 3

Modelos

- Modelo M_1:
  - Variáveis: $[j_1 \ldots j_p]$-w
- Número de variáveis é $\begin{pmatrix} p \times g \\ p \end{pmatrix} \times w$

Modelos

- Modelo M_2:
  - Variáveis: GOLFER_{ijkl}
    - i: jogador,
    - j: j-ésima posição no grupo,
    - k: grupo,
    - l: semana.
- Número de variáveis é $w \times (p \times g) \times g$.

- Cláusulas:

- Cada jogador joga no máximo uma vez por semana,
- Cada jogador joga pelo menos uma vez por semana,
- Cada jogador joga apenas num grupo em cada semana,
- Nenhum jogador joga no mesmo grupo que outro mais do que uma vez.

Resultados Experimentais

Resultados Experimentais

w-p-g	#Var	#Cls	M_1	#Var	#Cls	M_2
13-2-7	1183	21476	0.05	12103	86751	4.47
17-2-9	2601	62730	0.25	31671	242541	4.74
19-2-10	3610	97850	1.25	47500	372630	9.55
7-3-7	9310	3924312	4.47	14406	127302	0.51
5-4-4	9100	14123280	6.4	4000	35032	0.07

Round Robin

- Existem n equipas, com n par,
- Campeonato dura $n - 1$ semanas,
- Cada equipa joga um jogo por semana,
- Existem $\frac{n}{2}$ campos,
- Nenhuma equipa joga mais de duas vezes no mesmo campo.

Round Robin

Round Robin, $n = 6$

	Sem 1	Sem 2	Sem 3	Sem 4	Sem 5
Campo 1	1 2	2 3	1 5	4 6	3 5
Campo 2	3 4	1 6	3 6	2 5	1 4
Campo 3	5 6	4 5	2 4	1 3	2 6

Modelo

Variáveis

$\{p_{ij}^{1k} | 1 \leq i \leq n/2, 1 \leq j, k \leq n-1\} \cup$
$\{p_{ij}^{2k} | 1 \leq i \leq n/2, 1 \leq j \leq n-1, 2 \leq k \leq n\}$

- $n \times (n-1)^2$ variáveis,
- Cada local da matriz é preenchido por $(p_{ij}^{1k_1}, p_{ij}^{2k_2})$.

Modelo

Cláusulas

- Em cada local ocorre um jogo,
- Em cada local da forma $(p_{ij}^{1k}, p_{ij}^{2k'})$ temos que $k < k'$,
- Cada equipa joga um jogo em cada semana do campeonato,
- Cada par de equipas jogam entre si exactamente uma vez,
- Nenhuma equipa joga mais do que duas vezes no mesmo campo.

Simetrias do Problema

Round Robin, $n = 6$

	Sem 1	Sem 2	Sem 3	Sem 4	Sem 5
Campo 1	1 2	2 3	1 5	4 6	3 5
Campo 2	3 4	1 6	3 6	2 5	1 4
Campo 3	5 6	4 5	2 4	1 3	2 6

- Os jogadores são permutáveis,
- Jogadores no mesmo jogo são permutáveis,
- As semanas são permutáveis,
- Os campos são permutáveis.

Quebra de Simetrias

- Quebra de Simetrias de Semanas:
  - Fixar os jogos da forma (1,k),
  - Ordem lexicográfica no primeiro Campo.
- Quebra de Simetrias de Campos:
  - Fixar a primeira Semana,
  - Ordem lexicográfica na primeira Semana.

Quebra de Simetrias de Semanas

- Quebra de Simetrias de Semanas:
  - Fixar os jogos da forma (1,k),
  - Ordem lexicográfica no primeiro Campo.
- Quebra de Simetrias de Campos:
  - Fixar a primeira Semana,
  - Ordem lexicográfica na primeira Semana.

Quebra de Simetrias de Semanas

Fixar os jogos da forma (1,k)

	Sem 1	Sem 2	Sem 3	Sem 4	Sem 5
Campo 1	**1 2**	4 5	3 6	2 3	**1 6**
Campo 2	3 4	2 6	**1 4**	**1 5**	3 5
Campo 3	5 6	**1 3**	2 5	4 6	2 4

#Var	#Cls	#Sol	#Var(S)	#Cls(S)	#Sol(S)
150	2880	21600	162	2908	180

Quebra de Simetrias de Semanas

- Quebra de Simetrias de Semanas:
  - Fixar os jogos da forma (1,k),
  - Ordem lexicográfica no primeiro Campo.
- Quebra de Simetrias de Campos:
  - Fixar a primeira Semana,
  - Ordem lexicográfica na primeira Semana.

Quebra de Simetrias de Semanas

Ordem lexicográfica no primeiro Campo

	Sem 1	Sem 2	Sem 3	Sem 4	Sem 5
Campo 1	**1 2**	**1 3**	**2 5**	**3 5**	**4 6**
Campo 2	3 4	2 6	3 6	2 4	1 5
Campo 3	5 6	4 5	1 4	1 6	2 3

#Var	#Cls	#Sol	#Var(S)	#Cls(S)	#Sol(S)
150	2880	21600	150	5380	20

Quebra de Simetrias de Campos

- Quebra de Simetrias de Semanas:
  - Fixar os jogos da forma (1,k),
  - Ordem lexicográfica no primeiro Campo.
- Quebra de Simetrias de Campos:
  - Fixar a primeira Semana,
  - Ordem lexicográfica na primeira Semana.

Quebra de Simetrias de Campos

Fixar a primeira Semana

	Sem 1	Sem 2	Sem 3	Sem 4	Sem 5
Campo 1	**1 2**	2 3	1 5	4 6	3 5
Campo 2	**3 4**	1 6	3 6	2 5	1 4
Campo 3	**5 6**	4 5	2 4	1 3	2 6

#Var	#Cls	#Sol	#Var(S)	#Cls(S)	#Sol(S)
150	2880	21600	162	2886	240

Quebra de Simetrias de Campos

- Quebra de Simetrias de Semanas:
  - Fixar os jogos da forma (1,k),
  - Ordem lexicográfica no primeiro Campo.
- Quebra de Simetrias de Campos:
  - Fixar a primeira Semana,
  - Ordem lexicográfica na primeira Semana.

Quebra de Simetrias de Campos

Ordem lexicográfica na primeira Semana

	Sem 1	Sem 2	Sem 3	Sem 4	Sem 5
Campo 1	**1 2**	4 6	3 5	1 4	2 6
Campo 2	**3 6**	1 3	2 4	5 6	1 5
Campo 3	**4 5**	2 5	1 6	2 3	3 4

#Var	#Cls	#Sol	#Var(S)	#Cls(S)	#Sol(S)
150	2880	21600	162	2910	3600

Conclusões

- Conhecimento Geral dos Algoritmos SAT,
- Algoritmos SAT têm diversas aplicações,
- Algoritmos SAT estão bastante desenvolvidos,
- Importância da Codificação,
- Importância da quebra de Simetrias,
- Aplicação destes métodos ganha uma importância crescente.