Enunciado do Primeiro EP

Marcelo Finger

MAC239 — Segundo Semestre de 2013 — BCC/IME
Entrega: 23/09/2013

1 Objetivos

Os objetivos deste EP são que o estudante se familiarize com programas provadores de teoremas e a codificação automática de problemas em lógica proposicional.

2 Regras

1. Espera-se que este trabalho seja feito individualmente ou em duplas.
2. Para fins desta disciplina, as duplas contém exatamente 2 alunos, não existindo instâncias de duplas com número superior a 2 elementos ou número fracionário de elementos.
3. Cada indivíduo ou dupla entregará um relatório através do moodle no endereço `http://paca.ime.usp.br` pela conta de um dos membros da dupla. No caso das duplas, ambos devem estar inscritos no Paca e a nota será a mesma para ambos os membros da dupla de 2.
4. Deve ser entregue um único arquivo compactado (formato zip ou tgz), contendo o programa gerador das entradas para o resolvedor SAT e o relatório do experimento.
5. O programa deve ser escrito em uma das seguintes linguagens: C, C++, Java, Scala, Perl e Python. Só essas e mais nenhuma outra. Mesmo.
6. Descontar-se-á por erros ortográficos, gramaticais, desvios de estilo e outros impropérios à língua.

3 Sudokus

O problema que iremos resolver é um Sudoku.

O Sudoku é um quebra-cabeça cujo objetivo é preencher uma grade 9×9 com números de forma a nunca repetir um número em uma linha horizontal,

vertical ou na mesma região, onde uma região é uma subgrade 3×3. Veja abaixo um exemplo de Sudoku:

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

Figura 1: Configuração inicial de um Sudoku

Quando todos os números que faltam forem corretamente inseridos, temos uma configuração final válida o que caracteriza a vitória do jogador.

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

Figura 2: Configuração final de um Sudoku: resolvido

A idéia é codificar este problema como uma fórmula do cálculo proposicional, tal que cada valoração que satisfaz a fórmula representa uma disposição de números na grade.

Uma das formas de se fazer isso (mas nem de perto a única) é termos N^3 átomos da forma p_{ijk}, com $1 \leq i, j, k \leq 9$. Se tivermos uma valoração v tal que $v(p_{ijk}) = 1$, isto representa que na linha i e na coluna j do tabuleiro $N \times N$, temos o número k. Para uma valoração ser aceitável como solução do problema, ela deve satisfazer a exatamente N^2 átomos, e todo os demais $N^3 - N^2$ devem ser falsos.

Para a primeira parte do trabalho você deverá realizar os seguintes passos:

(a) Projetar uma fórmula que represente o problema do sudoku tal que uma valoração represente uma solução, conforme acima.

(b) Escrever um programa `sudoku` que recebe como entrada um arquivo com um sudoku incompleto e como saída imprime a fórmula que representa o problema.

O formato de entrada do sudoku consiste em 81 números separados por espaço que representam os números da posição $(1, 1)$ até a posição $(9, 9)$. O número 0 representa um espaço em branco. Por exemplo, o Sudoku representado na figura 1 é:

```
5 3 0 0 7 0 0 0 0 6 0 0 1 9 5 0 0 0 0 9 8 0 0 0 0 6 0 8 0 0 0 6
0 0 0 3 4 0 0 8 0 3 0 0 1 7 0 0 0 2 0 0 0 6 0 6 0 0 0 0 2 8 0 0
0 0 4 1 9 0 0 5 0 0 0 0 8 0 0 7 9
```

(c) A ativação do programa deve ser feita de forma clara, da forma

```
$ ./sudoku <arquivo_do_sudoku>
```

para gerar uma fórmula que representa o sudoku do arquivo.

Nota: esta fórmula deve estar na *forma clausal*, representada de acordo com as convenções do formato `cnf` descritas no documento presente no PACA.

Este formato é fundamental para que a fórmula seja aceita pelo resolvedor SAT.

4 O resolvedor SAT

Um resolvedor SAT é um programa que recebe uma fórmula proposicional e decide se ela é satisfazível ou não. Em geral, ao decidir se é satisfazível, ele também apresenta uma valoração.

Para nosso experimento iremos utilizar um resolvedor SAT de alta potência, o zchaff. O código fonte para este resolvedor pode ser obtido em

```
http://www.princeton.edu/~chaff/zchaff.html
```

Ao baixar o programa, você será solicitado a dizer as razões pelas quais você está baixando o programa. Você pode escrever o que quiser aí, mas o correto é dizer *coursework for an undergraduation course at the University of Sao Paulo.* Após baixar o pacote, você deve descompactá-lo e compilá-lo segundo as instruções, bastando usar o comando *make* no diretório extraído. Você precisará ter o compilador *g++* instalado.

Deve ser possível obter a resposta de um sudoku passando a fórmula obtida pelo seu programa para o zchaff.

O zchaff possui uma API C. Serão dados pontos de bônus se você utilizar essa API para chamar o zchaff internamente e responder com uma saída clara da solução do sudoku.