TRABALHO 1:
FAQ

RAUL H.C. LOPES

Seguem algumas questões que têm chegado a mim sobre o trabalho prático 1.

(1) **Como funciona a renomeação de variávies?**

A resolução binária só funciona (é consistente) se as cláusulas usadas não têm variáveis em comum. Assim, para as cláusulas C e D abaixo

$$C = \{p(?x, ?y), -q(?x, a), r(?x)\}$$
$$D = \{s(?u, ?x), -p(a, f(?y, ?x))\}$$

Devemos renomear todas as variáveis de D, garantindo que D e C não terão variáveis em comum. Os passos para isso poderiam ser:

(a) colete todas as variáveis de D: $\{?u, ?x, ?y\}$.
(b) a cada variável de D associe uma nova varivél, inexistente na prova: $\{(?u, ??0), (?x, ??1), (?y, ??2)\}$
(c) realize as substituições correspondentes em D, produzindo

$$\{s(??0, ??1), -p(a, f(??2, ??1))\}$$

Note que você precisará de um gerador de nomes de variáveis que produziria a cada chamada uma nova variável: $??_0$ na primeira vez, depois $??_1$, depois $??_2$, etc.

Por exemplo, a função *newvar* abaixo

```
newvar n = ("??"++show n):(newvar (n+1))
nv = newvar 0
```

produz uma lista infinita de strings representado nomes de variáveis. Use a função *head* aplicada a *nv* para recuperar o primeiro nome de variável: ??0; *tail* aplicado a *nv* prodizirá nova lista, cujo primeiro elemento será ??1.

(2) **Toda resolução binária usa uma cláusula do suporte?**

Sim e o resultado é incorporado ao suporte.

(3) **A teoria muda ao longo da prova?**

Se seu provador não usar estratégias para exclusão de cláusulas como eliminação de tautologias e de cláusulas subjugadas, a teoria não muda ao longo de um ramo da prova.

(4) **O que é o suporte inicialmente** Na especificação do trabalho propus a seguinte estratégia de busca: assumindo que seu conjunto representando a negação do teorema tem uma n cláusulas, você construirá n buscas diferentes, em cada uma o suporte inicial será formado por uma das n cláusulas que constituem a negação do teorema. A teoria será então a união das clásulas lidas do primeiro arquivo com as $n-1$ cláusulas restantes da negação do teorema. Você pode até restringir o suporte de forma que ele contenha a cada momento uma única cláusula: a última gerada por resolução binária ou fatoração. Isso produz uma busca que é chamada na lieteratura de resolução linear e é usada na linguagem Prolog.

(5) **Como implementar a busca da prova?**

Implemente seu provador como uma função que trabalha permanentemente sobre uma tripla, contendo:

- o conjunto de suporte;
- a teoria;
- uma lista de substituições.

Em um dado momento, o provador terá uma lista dessas triplas. Cada tripla representará um nó da árvore de busca. Selecione o primeiro nó e aplique inferências para produzir novas triplas. Cada inferência produzirá novo suporte, possivelmente nova teoria e atualizará a lista de substituições, produzindo novo nó a incorporar à árvore de pesquisa. Faça isso até encontrar a contradição.

(6) **Como obter as substituições a exibir na resposta final?** Ao encontrar a contradiçao, exiba a lista de substituições associada.

Por exemplo, suponha que sua teoria contém as cláusulas:

$$\{-p(?x), q(?x)\}$$
$$\{p(a)\}$$
$$\{p(b)\}$$

Assuma ainda que o teorema a provar é:

$$q(?x) \mid p(?y)$$

A sua negação produz:

$$\{-q(?x)\}$$
$$\{-p(?y)\}$$

Haverá duas árvores de busca da prova. Uma correspondendo ao conjunto de suporte inicial

$$\{-q(?x)\}$$

e a outra correspondendo ao conjunto

$$\{-p(?y)\}$$

Assumindo que Γ seja a união da teoria inicial com as cláusulas restantes do suporte, o primeiro suporte produzirá a seguinte seqüência de triplas, se assumirmos resolução linear:

$$(\{-q(?x)\}, \Gamma, \{\})$$
$$\langle \text{via cláusula1} \rangle$$
$$(\{-p(??0)\}, \Gamma, \{(?x, ??0)\})$$
$$\langle \text{via cláusula2} \rangle$$
$$(\{\}, \Gamma, \{(?x, a), (??0, a)\}$$

onde $\{\}$ representa a cláusula vazia, ou seja, contradição, e a resposta a exibir é

$$\{(?x, a), (??0, a)\}$$

ou mais corretamente

$$\{(?x, a)\}$$

dado que apenas $?x$ era parte da conjectura.