6
Conclusão

Bibliotecas de casamento de padrão são ferramentas presentes em diversas linguagens de programação. Muitas destas bibliotecas, baseadas em regexes, permitem a criação de padrões poderosos, graças ao seu conjunto expressivo de contruções. Porém, um ponto fraco dos regexes é a ausência de um modelo formal. Sem um modelo formal é difícil inferir a semântica de algumas construções e, consequentemente, o comportamento de algumas combinações.

Neste trabalho, apresentamos um estudo da conversão de regexes em PEGs. Como resultado deste estudo, criamos uma função que converte expressões regulares em PEGs baseada em continuação. Também criamos extensões desta função que permitem a conversão de algumas extensões de expressões regulares. Entretanto, não conseguimos propor conversões para algumas extensões de expressões regulares mais comuns, como backreferences, lookbehinds e âncoras de início. Ao analisarmos essas construções, percebemos que estas possuem uma característica em comum, que é depender de trechos que antecedem a posição atual do casamento. Esta característica explica a dificuldade em convertê-las para PEGs, já que as PEGs consideram que apenas o restante da entrada pode afetar o casamento.

Outro resultado deste trabalho foi mostrar que é possível executar regexes sobre implementações de PEGs e, com isso, oferecer uma base teórica para possíveis otimizações. Este resultado mostra que PEGs, através de um conjunto muito menor de construções, se comparado com regexes, proporcionam poder de expressão equivalente às construções mais comuns de regexes.

Na implementação do Lua Regex, fizemos uso da teoria produzida neste trabalho para construir uma biblioteca regex baseada em PEGs. Além disso, fizemos uma análise de desempenho que compara a nossa implementação com Perl, Lua e LPeg. Nesta análise não tivemos a intenção de mostrar que "*a implementação A é mais rápida que B*", e sim, procuramos mostrar como o modelo formal de PEGs nos permite checar possíveis otimizações que podem ser feitas durante a conversão para PEGs. Por exemplo, há casos em que as repetições de regexes podem ser convertidas para repetiçoes cegas de PEGs ao invés de utilizarmos recursões na gramática e, dessa forma, conseguimos casar

essas repetições de forma mais eficiente.

Uma linha de trabalho futuro consiste em tentar solucionar o problema da conversão de lookbehinds em PEGs. Na seção 4.6, propusemos um novo predicado sintático para PEGs, *back*, que consiste em retroceder uma posição do casamento e tentar casar uma dada expressão. Apesar de sua simplicidade aparente, a inserção deste predicado implica em mudanças no modelo conceitual de PEGs que, originalmente, assume que somente o restante da entrada pode afetar o resultado do casamento. Com a possibilidade de retroceder uma (ou mais) posições durante o casamento, é possível que este predicado proporcione mais poder de expressão às PEGs. Em contrapartida, novos problemas podem surgir, como loops infinitos. Assim, um estudo minucioso ainda deve ser feito em torno deste novo predicado de PEGs.