7
Conclusão

A ausência de uma definição formal para o conceito de co-rotinas e a diversidade de implementações de mecanismos de co-rotinas (algumas delas desnecessariamente complexas) impediram o reconhecimento da expressividade dessa construção de controle, e de sua conveniência como um recurso de programação.

Ao desenvolver este trabalho, buscamos alcançar um entendimento adequado do conceito de co-rotinas e de seu poder expressivo. Nossas principais contribuições são:

- a proposta de um novo sistema de classificação que permite distinguir diversas implementações de co-rotinas com respeito à sua conveniência e expressividade;
- a introdução do conceito de co-rotinas completas, e uma definição formal para esse conceito, baseada no desenvolvimento de uma semântica operacional;
- a demonstração da equivalência de poder expressivo entre co-rotinas completas simétricas e assimétricas e entre co-rotinas completas e continuações *one-shot*;
- uma discussão que fundamenta o argumento de que co-rotinas completas assimétricas são mais convenientes que co-rotinas completas simétricas para a implementação de diversos comportamentos de controle;
- uma comparação de modelos de concorrência com respeito a seus benefícios e desvantagens, justificando a adoção de modelos alternativos a *multithreading* e o oferecimento de co-rotinas como uma construção básica de concorrência adequada à implementação desses modelos alternativos.

A partir desses resultados, obtivemos também argumentos que fundamentam nossa defesa de co-rotinas como um conceito simples, adequado

a linguagens procedurais, que pode ser implementado com eficiência, e que pode substituir, com vantagens, tanto continuações de primeira classe quanto *threads*.

Atualmente, uma série de trabalhos vêm defendendo o uso de continuações de primeira classe para o desenvolvimento de aplicações WEB [72, 34, 26, 73]. A idéia básica dessas propostas é utilizar continuações de primeira classe para manter o estado de uma aplicação ao longo de um número arbitrário de interações com um cliente, permitindo que essa aplicação seja desenvolvida como uma aplicação sequencial convencional, ao invés de um conjunto de *scripts* isolados. O uso de continuações de primeira classe para a manutenção do estado das aplicações é, contudo, independente das implementações do modelo de concorrência para essas aplicações, que utilizam *threads*.

Acreditamos que o uso de co-rotinas nesse cenário é um campo de investigação bastante promissor. Além de prover um suporte conveniente para a manutenção do estado da aplicação WEB, preservando um estilo de programação sequencial, o uso de co-rotinas elimina a necessidade de construções de concorrência adicionais, como *threads*. Ao prover essas duas facilidades através de um único conceito, co-rotinas podem não apenas simplificar a estrutura da aplicação como também favorecer ganhos de desempenho.