Ferramenta educacional para auxiliar no ensino de Fork/Join

Lucas da Silva Marcos[1], **Rodolfo Adamshuk Silva**[1]

[1]Coordenação do Curso de Engenharia de Software
Universidade Tecnológica Federal do Paraná (UTFPR) – Dois Vizinhos, PR – Brazil

lucasmarcos@alunos.utfpr.edu.br, rodolfoa@utfpr.edu.br

Abstract.

Resumo.

1. Introdução

A programação concorrente é de extrema importância.

O ensino dela em universidade pode ser uma dificuldade.

Software de apoio ajudam na aprendizagem.

O modelo fork-join pode ser expresso em um linguagem de alto nível incrementada com as operações FORK e JOIN.

Uma analisador léxico e um analisador sintático apontam erros dos alunos no uso dessa linguagem.

Decomposição de tarefas em grafo de tarefas e métricas associadas.

A criação e extração dessas métricas.

Um portal web fácil aplicação em sala de aula. Uma pesquisa de resultados em sala de aula.

An increasing percentage of computation activity will be carried out by multiprogrammed computer systems. Such systems are characterized by the application of com- putation resources (processing capacity, main memory, file storage, peripheral equipment) to m a n y separate but concurrently operating computations.

1.1. Problema

Programação concorrente ou programação simultânea é um paradigma de programação para a construção de programas de computador que fazem uso da execução simultânea de várias tarefas computacionais interativas, que podem ser implementadas como programas separados ou como um conjunto de threads criadas por um único programa. Essas tarefas podem ser executadas por um único processador, vários processadores em um único equipamento ou processadores distribuídos por uma rede. Programação concorrente é relacionada com programação paralela, mas foca mais na interação entre as tarefas. A interação e a comunicação correta entre as diferentes tarefas, além da coordenação do acesso simultâneo aos recursos computacionais são as principais questões discutidas durante o desenvolvimento de sistemas simultâneos.

Vantagens do paradigma incluem o aumento de desempenho, pois aumenta-se a quantidade de tarefas sendo executadas em determinado período de tempo, e a possibilidade de uma melhor modelagem de programas, pois determinados problemas computacionais são simultâneos por natureza.

Ensino de programação concorrente.

Ensino de programação no ensino superior.

Unidade curricular de Programação Concorrente e Distribuída.

Na computação paralela, o modelo fork-join (bifurcar-juntar) é uma forma de configurar e executar programas em paralelo de tal forma que a execução das tarefas concorrentes "juntam-se"(join) em um ponto subsequente, onde a execução passa a ser sequencial. As seções paralelas podem se bifurcar recursivamente até que uma determinada granularidade da tarefa seja atingida. Fork-join pode ser considerado como um padrão de projeto paralelo. Foi formulado já em 1963.

Algumas implementações.

Decomposição. Grafo de tarefas. Métricas.

Linguagem de alto nível.

Um compilador é um programa de computador que, a partir de um código fonte escrito em uma linguagem compilada, cria um programa semanticamente equivalente, porém escrito em outra linguagem, código objeto.

Na ciência da computação, análise léxica, lexing ou tokenização é o processo de converter uma sequência de caracteres (como em um programa de computador ou página da web) em uma sequência de tokens (strings com um significado atribuído e, portanto, identificado). Um programa que realiza análise lexical pode ser denominado lexer, tokenizer, ou scanner, embora scanner também seja um termo para o primeiro estágio de um lexer. Um lexer geralmente é combinado com um analisador, que juntos analisam a sintaxe de linguagens de programação, páginas da Web e assim por diante.

Em ciência da computação e linguística, a análise sintática (do inglês: parsing) é um processo de um compilador (de uma linguagem de programação), é a segunda fase da compilação onde se analisa uma sequência que foi dada entrada (via um arquivo de computador ou via teclado, por exemplo) para verificar sua estrutura gramatical segundo uma determinada gramática formal. Este processo trabalha em conjunto com a análise lexical (primeira etapa, onde se verifica de acordo com determinado alfabeto) e análise semântica (terceira etapa, onde verificam-se os erros semânticos).

A análise sintática transforma um texto na entrada em uma estrutura de dados, em geral uma árvore, o que é conveniente para processamento posterior e captura a hierarquia implícita desta entrada. Através da análise lexical é obtido um grupo de tokens, para que o analisador sintático use um conjunto de regras para construir uma árvore sintática da estrutura.

Em termos práticos, por exemplo, pode também ser usada para decompor

um texto em unidades estruturais para serem organizadas dentro de um bloco.

A vasta maioria dos analisadores sintáticos implementados em compiladores aceitam alguma linguagem livre de contexto para fazer a análise. Estes analisadores podem ser de vários tipos, como o LL, LR e SLR.

1.2. Motivação

Conceitos complexos de programação concorrente.

Escassez de material e ferramentas de apoio e suporte a educação.

Dificuldades de aprendizado.

A semântica é definida para várias meta-instruções que realizam operações essenciais para a escrita de programas em sistemas de computadores multiprogramados. Essas meta-instruções estão relacionadas ao processamento paralelo, proteção de cálculos separados, depuração de programas e o compartilhamento entre usuários de segmentos de memória e outros objetos de computação, cujos nomes são estruturados hierarquicamente. A sofisticação da linguagem contemplada está no meio do caminho entre uma linguagem de montagem e uma linguagem algébrica avançada.

Sabe-se que o modelo Fork/Join foi criado nos anos 60, sendo esse a base de quase todos os modelos de concorrência usados atualmente. Todavia, ainda é um conteúdo que apresenta pouca referência na literatura científica, uma vez que é usado de modo muito limitado, apresentando-se quase que exclusivamente apenas nas disciplinas de Programação Concorrente Distribuída.

Dessa forma, objetivo desta pesquisa é desenvolver uma ferramenta educacional para auxiliar no ensino do modelo Fork/Join de concorrência no contexto de uma unidade curricular para cursos de Ensino Superior em Engenharia de Software com foco nas atividades desenvolvidas na disciplina de Programação Concorrente Distribuída.

Considerando o fato de que o assunto é pouco descrito em pesquisas científicas, como já citado, esta pesquisa traz como diferencial o objetivo de desenvolver uma ferramenta educacional para auxiliar professores e alunos durante o processo de ensino-aprendizagem do modelo Fork/Join ao longo das aulas de Programação Concorrente Distribuída dos cursos de Engenharia de Software.

Assim, a necessidade dessa ferramenta evidencia-se ao percebermos que os conceitos de programação concorrente têm como principal limitação a pouca variedade de ferramentas de suporte. Fato esse que dificulta no uso do modelo Fork/Join como também desenvolver outras atividades relacionadas à área.

Nesse sentido, pretende-se desenvolver a ferramenta por meio de um compilador para uma linguagem de alto nível que represente o modelo Fork/Join e que auxilie na correção de sintaxe, criação do grafo de paralelismo e exiba alguns tipos de métricas (das métricas, essas só serão possíveis termos conhecimento de quais tipos serão alcançados após o desenvolvimento da ferramenta).

Portanto, o estudo para o desenvolvimento desta pesquisa e, também, dessa possível ferramenta serão baseados em artigos sobre Fork/Join e compiladores para desenvolver um artigo apresentando um compilador de apoio ao ensino.

1.3. Objetivo

Construção de uma ferramenta de apoio ao ensino por meio de um compilador para uma linguagem de alto nível que represente o modelo fork-join.

Desenvolver uma ferramenta educacional para auxiliar no ensino do modelo Fork/Join de concorrência no contexto de uma unidade curricular para cursos de Ensino Superior em Engenharia de Software com foco nas atividades desenvolvidas na disciplina de Programação Concorrente Distribuída.

O objetivo desta pesquisa é desenvolver uma ferramenta educacional para auxiliar no ensino do modelo Fork/Join de concorrência no contexto de uma unidade curricular para cursos de Ensino Superior em Engenharia de Software com foco nas atividades desenvolvidas na disciplina de Programação Concorrente e Distribuída.

Considerando o fato de que o assunto é pouco descrito em pesquisas científicas, como já citado, esta pesquisa traz como diferencial o objetivo de desenvolver uma ferramenta educacional para auxiliar professores e alunos durante o processo de ensino-aprendizagem do modelo Fork/Join ao longo das aulas de Programação Concorrente e Distribuída dos cursos de Engenharia de Software.

Assim, a necessidade dessa ferramenta evidencia-se ao percebermos que os conceitos de programação concorrente têm como principal limitação a pouca variedade de ferramentas de suporte.

Fato esse que dificulta no uso do modelo Fork/Join como também desenvolver outras atividades relacionadas à área.

Nesse sentido, pretende-se desenvolver a ferramenta por meio de um compilador para uma linguagem de alto nível que represente o modelo Fork/Join e que auxilie na correção de sintaxe, criação do grafo de paralelismo e exiba alguns tipos de métricas (das métricas, essas só serão possíveis termos conhecimento de quais tipos serão alcançados após o desenvolvimento da ferramenta).

Portanto, o estudo para o desenvolvimento desta pesquisa e, também, dessa possível ferramenta serão baseados em artigos sobre Fork/Join e compiladores para desenvolver um artigo apresentando um compilador de apoio ao ensino.

2. Trabalhos relacionados

Fork/Join [Conway 1963] e [Dennis and Van Horn 1966].

Sistemas Distribuídos [Coulouris et al. 2001] e [Van Steen and Tanenbaum 2017].

Compiladores [Louden 2004].

Ferramentas educacionais.

Materia de apoio para aprendizagem de Programação Concorrente.

Compiladores em sala de aula.

Compiladores de Pascal usados para o ensino de matemática.

3. Proposta

O desenvolvimento de um compilador e plataforma com o foco no uso em sala de aula.

Será especificado uma gramática formal da linguagem de alto nível utilizada. A linguagem possuirá os primitivos Fork e Join como modelo de concorrência.

Será construído um analisador léxico, reportando aos alunos erros nos programas.

Será construído um analisador sintático, que modelara a estrutura do programa e reportará erros de construção.

Como saída esperada teremos, além das mensagens de erro ou de sucesso, um grafo de tarefas e dependências, além de métricas do programa especificado, como o caminho crítico e o grau máximo de concorrência.

Também será construído uma plataforma web, como um editor de textos integrado e renderização do grafo e métricas. Exemplo Compiler Explorer https://godbolt.org.

4. Metodologia

Pesquisa aplicada e exploratória.

Aplicaremos o protótipo em sala de aula, coletaremos dados e feedback dos alunos. Feedback pesquisa qualitativa.

Cronograma.

5. Resultados esperados

Espera-se que a ferramenta proposta contribua significativamente para o aprimoramento do ensino de programação concorrente, fornecendo um recurso para o aprendizado dos alunos e auxiliando os professores no processo de ensino.

Referências

Conway, M. E. (1963). A multiprocessor system design. In *Proceedings of the November 12-14, 1963, fall joint computer conference*, pages 139–146.

Coulouris, G., Dollimore, J., and Kindberg, T. (2001). *Sistemas distribuidos*, volume 6. Addison Wesley Madrid.

Dennis, J. B. and Van Horn, E. C. (1966). Programming semantics for multiprogrammed computations. *Communications of the ACM*, 9(3):143–155.

Louden, K. (2004). *Compiladores - Princípios e Práticas*. Pioneira Thomson Learning.

Van Steen, M. and Tanenbaum, A. S. (2017). *Distributed systems*. Maarten van Steen Leiden, The Netherlands.