

UNIVERSIDADE FEDERAL FLUMINENSE  
INSTITUTO DE COMPUTAÇÃO  
DEPARTAMENTO DE CIÊNCIA DA COMPUTAÇÃO

**Luiz Laerte Nunes da Silva Junior**  
**Thiago Nazareth de Oliveira**

## **Vertical Code Completion**

Niterói  
2010

**Luiz Laerte Nunes da Silva Junior**

**Thiago Nazareth de Oliveira**

## **Vertical Code Completion**

**Monografia apresentada ao Departamento de Ciência da Computação da Universidade Federal Fluminense como parte dos requisitos para obtenção do Grau de Bacharel em Ciência da Computação.**

Orientadores: Leonardo Murta

Co-orientador: Alexandre Plastino

Niterói

2010

**Luiz Laerte Nunes da Silva Junior**

**Thiago Nazareth de Oliveira**

## **Vertical Code Completion**

**Monografia apresentada ao Departamento de Ciência da Computação da Universidade Federal Fluminense como parte dos requisitos para obtenção do Grau de Bacharel em Ciência da Computação.**

Aprovado em Junho de 2010

### **BANCA EXAMINADORA**

---

Prof. Leonardo Murta, D.Sc.  
Orientador  
UFF

---

Prof. Alexandre Plastino, D.Sc.  
Co-Orientador  
UFF

---

Prof. - , M.Sc.  
UFF

---

Prof. - , D.Sc.  
UFF

Niterói  
2010

# RESUMO

VCC VERTICAL CODE COMPLETION AQUI ENTRA O RESUMO

**Palavras Chave:**

Engenharia de Software, Code Completion, Mineração de Dados, Mineração de Padrões Sequenciais.

# ABSTRACT

VCC VERTICAL CODE COMPLETION AQUI ENTRA O ABSTRACT

**Keywords:**

Software Engineering, Code Completion, Data Mining, Sequence Mining.

# LISTA DE ACRÔNIMOS

VCC: *Vertical Code Completion*

# SUMÁRIO

<b>CAPÍTULO 1 - INTRODUÇÃO</b>	<b>5</b>
<b>CAPÍTULO 2 - REVISÃO DA LITERATURA</b>	<b>6</b>
2.1 Introdução . . . . .	6
2.2 Mineração de Dados . . . . .	6
2.2.1 Tarefas de Mineração de dados . . . . .	6
2.2.1.1 Classificação . . . . .	7
2.2.1.2 Regras de Associação . . . . .	7
2.2.1.3 Clusterização . . . . .	7
2.2.1.4 Padrões em Séries Temporais . . . . .	8
2.2.1.5 Padrões Sequenciais . . . . .	8
2.3 Mineração de dados na Engenharia de Software . . . . .	9
<b>CAPÍTULO 3 - VERTICAL CODE COMPLETION</b>	<b>11</b>
<b>CAPÍTULO 4 - IMPLEMENTAÇÃO</b>	<b>12</b>
<b>CAPÍTULO 5 - RESULTADOS EXPERIMENTAIS</b>	<b>13</b>
<b>CAPÍTULO 6 - CONCLUSÕES</b>	<b>14</b>
<b>REFERÊNCIAS BIBLIOGRÁFICAS</b>	<b>15</b>

## **LISTA DE FIGURAS**



## **LISTA DE TABELAS**

# **CAPÍTULO 1 - INTRODUÇÃO**

Capítulo 1

## **CAPÍTULO 2 - REVISÃO DA LITERATURA**

### **2.1 INTRODUÇÃO**

Neste capítulo são abordados alguns conceitos e técnicas de Mineração de Dados, além da aplicação dessas técnicas em problemas de Engenharia de Software, citando trabalhos desenvolvidos nessa área.

### **2.2 MINERAÇÃO DE DADOS**

A evolução computacional das últimas décadas, ocasionada pela evolução tecnológica, permitiu um grande aumento no poder de processamento e na capacidade de armazenamento de dados a baixo custo, inserindo no mercado, novas tecnologias de transmissão e disponibilização de dados[3]. Isso permitiu que empresas e centros de pesquisa acumulassem grandes quantidades de dados históricos a partir dos anos 70 e 80[5].

Porém, essa enorme quantidade de dados não refletia uma grande riqueza de conhecimento, pois não era analisada com ferramentas adequadas, já que um volume tão extenso de informações ultrapassa a habilidade humana de compreensão. Consequentemente, importantes decisões eram frequentemente tomadas somente por intuição, simplesmente pela falta dessas ferramentas que poderiam extrair conhecimentos valiosos dos repositórios de dados [5].

Isso motivou diversos estudos a partir do início dos anos 90, que resultaram na criação do campo de pesquisa de Mineração de Dados, área que se refere ao processo de descoberta de novas informações e conhecimento, no formato de regras e padrões, a partir de grandes bases de dados [7]. A partir daí, foram desenvolvidas ferramentas para analisar e descobrir importantes padrões de dados, contribuindo para diversas áreas, tais como [5]: pesquisas médicas, negócios estratégicos, biologia molecular, entre outras.

#### **2.2.1 TAREFAS DE MINERAÇÃO DE DADOS**

Existem diversas tarefas em Mineração de Dados, as principais [4], que serão apresentadas nesse capítulo são: classificação, regras de associação, clusterização, padrões em séries tempo-

rais e padrões seqüências. Em geral, essas tarefas podem ser classificadas em duas categorias: mineração preditiva e mineração descritiva [5].

Na mineração preditiva, deseja-se prever o valor desconhecido de um determinado atributo, a partir da análise histórica dos dados armazenados na base. Nessa categoria se enquadram as tarefas de classificação e padrões em séries temporais. Na mineração descritiva, padrões e regras descrevem características importantes dos dados dos quais se está trabalhando. As tarefas de regras de associação, clusterização e padrões sequenciais fazem parte dessa categoria.

### **2.2.1.1 CLASSIFICAÇÃO**

A tarefa de classificação tem por objetivo identificar, entre um conjunto pré-definido de classes, aquela a qual pertence um elemento a partir de seus atributos. Para inferir a qual classe esse elemento pertence, é necessária uma base de treinamento.

Um sistema de um banco, que tem por objetivo inferir se um cliente será ou não um bom pagador, com base nos dados de clientes antigos e nas características do elemento que está sendo classificado, é um exemplo de utilização da tarefa de classificação.

### **2.2.1.2 REGRAS DE ASSOCIAÇÃO**

Uma regra de associação representa um padrão de relacionamento entre itens de dados do domínio da aplicação, que ocorre com uma determinada frequência. Essas regras são extraídas a partir de uma base de dados organizada em transações, que são formadas por um conjunto de itens desse domínio. Um exemplo fictício de regra de associação que poderia ser extraído de uma base de dados de vendas é: clientes que compram o produto A geralmente compram o produto B.

Outro exemplo, porém real, que foi extraído de uma base de dados médica após a aplicação dessa técnica, descreve que pacientes aidéticos que contraem a doença candidíase também têm pneumonia.

### **2.2.1.3 CLUSTERIZAÇÃO**

A tarefa de clusterização é usada para agrupar elementos de uma base de dados através de seus atributos ou características, de forma que elementos mais similares fiquem no mesmo cluster e elementos menos similares fiquem em clusters distintos.

Esta técnica é muito utilizada em sistemas de grandes operadoras de cartão de crédito,

separando os clientes em grupos de forma que aqueles que apresentam o mesmo comportamento de consumo fiquem no mesmo grupo. A separação desses clientes por grupo pode ser usada para fazer algum tipo de marketing apropriado ao grupo ou na detecção de fraudes, no caso de um cliente apresentar um comportamento diferente do esperado para o seu perfil.

#### 2.2.1.4 PADRÕES EM SÉRIES TEMPORAIS

Consiste de uma sequência de valores mensurados em intervalos iguais de tempo [5]. O principal objetivo da análise de padrões sequenciais em séries temporais é realizar previsões futuras baseando-se no histórico dos dados.

Cotação diária do dólar, faturamento anual de uma empresa e evolução do índice da bolsa de valores são exemplos de séries temporais.

#### 2.2.1.5 PADRÕES SEQUENCIAIS

Nesta seção detalharemos com maior profundidade a técnica de extração de padrões sequenciais, visto que essa será aplicada em nosso trabalho.

Existem muitas aplicações envolvendo dados sequenciais, e a ordem com que esses dados aparecem é muito importante para análise e entendimento de alguns padrões. Exemplos típicos incluem sequências de compras de um cliente, sequências biológicas e sequências de eventos na ciência e na engenharia. Padrões sequenciais representam sequências de eventos ordenados, que aparecem com significativa frequência em uma base de dados. Um exemplo de padrão sequencial é: "clientes que compram uma câmera digital Canon comumente compram uma impressora HP colorida dentro de um mês".

Uma sequência é uma lista ordenada de eventos e seu tamanho é determinado pelo seu número de itens. Uma sequência  $s$  é representada por  $\langle e_1 e_2 e_3 \dots e_n \rangle$ , onde  $e_k$  é dito um evento ou elemento da sequência  $s$  e  $e_1$  ocorre antes de  $e_2$ , que ocorre antes de  $e_3$  e assim sucessivamente. Por sua vez, um evento ou elemento da sequência é representado por  $e = (i_1 i_2 i_3 \dots i_m)$ , onde  $i_k$  é um item do domínio da aplicação.

Podemos dizer que um evento em uma base de dados de uma loja de vendas é uma compra feita por um cliente, e o item do domínio da aplicação são os produtos que pertencem à essa compra.

Além disso, outras definições são importantes. Uma sequência pode ser parte de outra sequência maior. Nesse caso, a sequência  $\alpha = \langle a_1 a_2 \dots a_n \rangle$  é chamada de subsequência de

outra sequência  $\beta = \langle b_1 b_2 \dots b_m \rangle$ , e  $\beta$  é uma supersequência de  $\alpha$ , denotado como  $\alpha \subseteq \beta$ , se existirem inteiros  $1 \leq j_1 < j_2 < \dots < j_n \leq m$  tais que  $a_1 \subseteq b_{j_1}, a_2 \subseteq b_{j_2}, \dots, a_n \subseteq b_{j_n}$ . Por exemplo, se  $\alpha = \langle (ab), d \rangle$  e  $\beta = \langle (abc), (de) \rangle$ , onde  $a, b, c, d$  e  $e$  são itens, então  $\alpha$  é uma subsequência de  $\beta$  e  $\beta$  é uma supersequência de  $\alpha$ .

Entretanto, padrões sequenciais se tornam inúteis se não pudermos classificá-los corretamente. Nesse contexto o suporte se apresenta como métrica de avaliação. Podemos definir o suporte de uma sequência como a quantidade de vezes que a mesma ocorre em determinada base de dados. Este suporte pode ser absoluto, representado apenas por um número inteiro, ou relativo, informando o percentual de vezes que a mesma sequência ocorreu. Dessa forma, uma sequência de eventos é dita frequente se a quantidade de vezes que essa sequência ocorrer for superior ao suporte mínimo, formando assim um padrão sequencial.

## 2.3 MINERAÇÃO DE DADOS NA ENGENHARIA DE SOFTWARE

Tarefas de mineração de dados têm sido muito utilizadas na engenharia de software, principalmente aplicando suas técnicas em repositórios de dados para obter informações sobre a evolução de software ao longo do tempo, com o objetivo de aumentar a qualidade do processo de desenvolvimento de software [6, 8, 2, 9, 1].

[6] utilizou um algoritmo classificador, a partir do aprendizado em uma base de treinamento, para classificar a relação de relevância de manutenção entre dois arquivos. Esta relação é utilizada no contexto de manutenção de software.

Em seu trabalho, [8] aplica mineração de dados para encontrar relações de dependências no código fonte para ajudar os desenvolvedores em tarefas de modificação. Em especial, é utilizada a técnica de extração de regras de associação para determinar padrões de mudanças no código fonte.

[2] utiliza técnicas de mineração de dados num repositório UML versionado para extrair regras de associação que possam identificar elementos do modelo UML que foram modificados juntos no passado e que provavelmente precisarão ser modificados juntos no futuro.

[9] aplica técnicas de mineração de dados em repositórios versionados de código fonte para extrair regras de associação do tipo: "programadores que alteraram a função A também alteraram as funções B e C". Tais regras são extraídas com o objetivo de guiar desenvolvedores de software na alteração do código fonte.

Em [1], um algoritmo de clusterização foi utilizado num sistema de controle de versões

para identificar classes semanticamente relacionadas. A partir do gráfico gerado pelo algoritmo, pôde-se analisar que mudanças em classes de certo cluster eram frequentemente envolvidas com mudanças em classes de outro cluster.

## **CAPÍTULO 3 - VERTICAL CODE COMPLETION**

Cap 3.



## **CAPÍTULO 4 - IMPLEMENTAÇÃO**

### Capítulo 4

## **CAPÍTULO 5 - RESULTADOS EXPERIMENTAIS**

Capítulo 5

## **CAPÍTULO 6 - CONCLUSÕES**

Capítulo 6

## REFERÊNCIAS BIBLIOGRÁFICAS

- [1] T. Ball, J. Kim, A.A. Porter e et al. If your version control system could talk. *Workshop on Process Modelling and Empirical Studies of Software Engineering*, 1, 1997.
- [2] C. Dantas, L. Murta e C. Werner. Mining change traces from versioned uml repositories. *XXI Simpósio Brasileiro de Engenharia de Software (SBES 2007)*, 1, October de 2007.
- [3] M. A. Domingues. Generalização de regras de associação. Dissertação de Mestrado, USP - São Carlos, 2004.
- [4] E.C. Gonçalves. Regras de associação e suas medidas de interesse objetivas e subjetivas. *INFOCOMP Journal Computer Science*, 1, 1999.
- [5] J. Han e M. Kamber. *Data Mining: Concepts and Techniques (2nd edition)*. Morgan Kaufmann, 2006.
- [6] J.S. Shirabad, T. Lethbridge e S. Matwin. Supporting software maintenance by mining software update records. *International Conference on Software Maintenance (ICSM)*, 1:22–31, November de 2001.
- [7] R. Srikant e R. Agrawal. Mining sequential patterns: Generalizations and performance improvements. Em *Proceedings of the 5th International Conference on Extending Database Technology - EDBT*, 1996.
- [8] A.T.T. Ying, G.C. Murphy, R. Ng e et al. Predicting source code changes by mining change history. *IEEE Transactions on Software Engineering (TSE)*, 30(9):574–586, 2004.
- [9] T Zimmermann, P. Weisgerber, S. Diehl e et al. Mining version histories to guide software changes. *International Conference on Software Engineering (ICSE)*, 1:563–572, May de 2004.