Métricas de Software Aplicadas a Gramáticas

Mestrado em Engenharia Informática - EL
Universidade do Minho

Bruno Costa
pg17778@alunos.uminho.pt

Nuno Oliveira
pg18391@alunos.uminho.pt

1 Introduçao

1.1 Métricas de software

As métricas de *software* permitem quantificar certas propriedades associadas a uma aplicação, estas propriedades podem ser divididas em dois grandes grupos, aquelas que se aplicam a todo o processo de especificação e aquelas que se aplicam a implementação da mesma. De uma forma geral as métricas que se aplicam ao primeiro grupo requerem intervenção humana no seu cálculo, ou seja, o seu cálculo nao pode ser totalmente automatizado. Por outro lado, o cálculo das métricas associadas à implementação pode, na sua grande parte, ser automatizado. Neste documento vamos focar-nos nas aplicações *grammar-based*, por exemplo compiladores. Uma parte fundamental destas aplicações é a definição da gramática que lhes está associada, portanto, quando avaliamos a qualidade de uma dessas aplicações não podemos deixar de lado essa gramática. Embora sejam escritas na forma de código as gramáticas não deixam de ser especificações, portanto as métricas *standard* não se lhes aplicam. Portanto, neste contexto, ao longo deste documento vamos descrever as métricas que existem para avaliar a qualidade de gramáticas.

1.2 Métricas aplicadas a especificações

A especificação é o ponto de partida para a construção de *software*, portanto a sua qualidade é de extrema importância. Logo é necessário possuir métodos que nos permitam avaliar, de uma forma uniformizada, a sua qualidade. Davis em 1993 propôs um conjunto de métricas para estudar a qualidade da especificação de *software*. Estas métricas embora não possam, de uma forma geral, ser calculadas de forma completamente automática, permitem automatizar e uniformizar o processo de avaliação de especificações. Dessas métricas destacam-se as seguintes:

- Falta de Ambiguidade
- *Completeness*
- Consistência
- *Correctness*
- Capacidade de Compreensão
- Verificabilidade
- *Achievability*
- Concisão
- *Traceability*
- Modificabilidade
- Precisão
- Reutilizabilidade

Algumas das métricas enunciadas em cima podem ser calculadas da seguinte forma:

O número de requisitos:

$$n_r = n_f + n_{nf}$$

onde n_r é o número total de requisitos, n_f requisitos funcionais e n_{nf} requisitos não funcionais.

A falta de ambiguidade é calculada através da seguinte fórmula:

$$Q_1 = n_{ui}/n_r$$

onde n_{ui} é o número de requerimentos que foi interpretados da mesma maneira por todos os *reviewers* (n_r).

A *Completeness* é obtida utilizando a fórmula:

$$Q_2 = n_u/[n_i \times n_s]$$

onde n_u é os requerimentos funcionais únicos,n_i número de *inputs* e n_s – numero de estados especificados.

2 Métricas para gramáticas

No estudo que fizemos, encontrámos várias métricas que podem ser aplicadas a gramáticas. Estas métricas podem ser divididas em dois grupos; aquelas que derivam das métricas *standard* que são aplicadas ao *software* e aquelas que são próprias às gramáticas. Estes grupos são designados, respectivamente, por *Métricas de Tamanho* e *Métricas de Estrutura*.

As *Métricas de Tamanho* sobre as quais nos debruçámos foram as seguintes:

- Número de terminais e não-terminais
- Complexidade Ciclomática de McCabe
- Tamanho médio de RHS (*Right Hand Side*)
- *Halstead effort*

No que diz respeito ás *Métricas de Estrutura*, analisámos as seguintes:

- Impureza
- Contagem dos níveis normalizados
- Número de níveis *non-singleton*
- Tamanho do maior nível
- Altura máxima

Vamos agora descrever de uma forma mais pormenorizada cada uma destas métricas.

2.1 Métricas de Tamanho

Como foi dito estas métricas derivam das regras *standard* que estão associadas ao *software* em geral. Vamos agora para cada uma descrever a sua forma de cálculo e uma interpretação possível para o seu valor.

2.1.1 Número de terminais e nao-terminais

A métrica mais simples que se pode aplicar é a contagem de não-terminais, geralmente esta pode ser devolvida de forma automática geradores de *parsers* como o *YACC*, por exemplo. Um elevado número de não-terminais implica maior custo de manutenção, pois alterações na definição de um pode ter impacto em muitos outros. O tamanho da *parse table* geralmente é proporcional ao número de terminais e não-terminais, particularmente em algoritmos *LL* e *LALR*, o que influencia a a eficiência do *parser* gerado.

2.1.2 Complexidade Ciclomática de McCabe

A *CCM* mede o número de caminhos lineares independentes num grafo de dependências Interpretada com a medida do número de decisões no grafo, onde as decisões são representadas pelo uso de expressões de valores *booleanos* em iterações e condições. Para aplicar isto em gramáticas, temos de contar o número total de alternativas nessa gramática, representadas por ocorrências de operadores de opção ou de união (*yacc* e *bison* apenas utilizam o operador de união)). Como o objectivo de um *parsing algorythm* é providenciar um meio de escolher entre alternativas numa gramática durante uma derivação, esta métrica indica um maior potencial para conflitos(quando reporta um valor alto) num *parser lookahead*, e grande *scope* para *backtracking* num *search-based parser*.

2.1.3 Tamanho médio de RHS (*Right Hand Side*)

Para uma função, esta métrica mede o número de nodos no grafo correspondente ao número de vezes usada como alternativa comparado com o número de linhas de código. As produções de uma gramática são equivalentes às funções, os nodos no grafo correspondem, portanto a terminais ou não-terminais no *RHS* de uma regra de produção. Este é obtido a partir do total dos tamanhos do *RHS* para cada regra e dividido pelo número de não-terminais. Com esta análise podemos obter o número médio de símbolos que poderão aparecer no lado direito de uma regra de gramática. Em alguns *parsers*, maior *RHS* pode significar um maior número de símbolos ou atributos a serem colocados na *stack* do *parser*, o que diminuirá a eficiência do mesmo.

2.1.4 *Halstead effort*

Halstead define duas métricas para quantificar programas:

- V medida do tamanho do programa;
- E a estimativa do esforço necessário para entender o programa;

Estas métricas são calculadas em função do número de funções, operadores e operandos de um programa. Para as implementar em gramáticas, interpretamos os operadores como as produções gramaticais e os operandos como os terminais e não-terminais de uma determinada gramática. O valor de V não trás mais informação que a métrica que calcula o número de terminais e não-terminais, no entanto o E, relativiza a *Complexidade Ciclomática de McCabe*, pois conta o número de operadores através da multiplicação de um peso pelo número de ocorrências dos símbolos da gramática. Esta abordagem permite uma melhor medida das diferenças de complexidade entre gramáticas de tamanhos diferentes.

2.2 Métricas de Estrutura

Métricas de Estrutura são métricas que dão informação acerca da estrutura em grafo dada pela gramática.

2.2.1 Impureza

Esta análise mede quanto o grafo se assemelha a uma árvore (0% este assemelha-se a uma árvore e 100% este tem os nodos todos interligados). Com um valor alto, ao fazer um *refactoring* da árvore vamos ter complicações, visto que a alteração de uma regra poderá ter impacto em outras.

2.2.2 Contagem dos níveis normalizados

Esta contagem é a normalização do número de níveis gramaticais pelo número de não-terminais expresso em percentagem. Um elevado número indica mais oportunidades para modularização da gramática.

2.2.3 Número de níveis *non-singleton*

Esta métrica apenas contabiliza o número classes non-singleton.

2.2.4 Tamanho do maior nível

A métrica de profundidade para uma gramática mede o número de não-terminais no maior nível gramatical. Caso o valor do tamanho seja uma grande parte do número total de não-terminais, significa que existe uma distribuição desequilibrada dentro destes níveis.

2.2.5 Altura máxima (*Varju Height metrics*)

A altura máxima é a distância máxima de qualquer não-terminal até ao Símbolo Inicial, e é expressa como uma percentagem do número de classes equivalentes.

3 Novas métricas propostas

Nesta secção vamos analisar dois novos tipos de métricas, Métricas baseadas em tabelas LR e Métricas baseadas em linguagem gerada.

3.1 Métricas baseadas em tabelas LR

Estas métricas são baseadas num autómato LR que é utilizado para produzir *parsers bottom-up* eficientes. A partir deste autómato, podemos extrair as seguintes métricas:

- *lrs* que representa o número de estados do autómato. A partir desta métrica podemos obter a complexidade da gramática.
- *lat* calcula dado um terminal, a média de estados no autómato LR que pode alterar este terminal. Esta métrica dá o valor da probabilidade de um terminal ser alterado durante o *parsing*.
- *lrtla* calcula a média de terminais que podem ser alterados em cada estado, dando então a complexidade de cada estado.

3.2 Métricas baseadas em linguagem gerada

A segunda proposta é baseada nas propriedades da linguagem gerada. As diferentes métricas apresentadas são as seguintes:

- ss
- ssm
- ltps
- ltpsm
- ltpsa
- ltpsn

Por *ss* entendemos como o tamanho médio das *samples* mais pequenas contidas numa dada produção. O *ssm* é o tamanho máximo da *sample*. Os restantes itens apenas podem ser aplicados a pares de terminais. *Ltps* calcula o número de diferentes pares de terminais aceitados na linguagem. *Ltpsm* calcula o número máximo de diferentes pares para um terminal. *Ltpsa* permite calcular, dado um terminal, a média de terminais que directamente podem seguir este dado terminal. *Ltpsn* normaliza as métricas *lpts* com o número de combinações possíveis de terminais (o valor é expresso em percentagem).

4 Conclusão

Métricas de *software* são sem qualquer dúvida úteis, no entanto a análise que estas permitem fazer quando aplicadas a especificações ou a gramáticas, a sua análise é bastante superficial. Para obter informação mais detalhada, é necessário recorrer a métricas específicas. Apesar das métricas de especificações (propostas por Davis) não serem automatizadas em toda a sua totalidade e os resultados dependerem da compreensão de humanos, são uma boa ferramenta para estabelecer bases com que construir o *software*. As métricas para gramáticas estão divididas em duas categorias, as de tamanho e as de estrutura. As de tamanho são uma aplicação das métricas para *software* sobre uma gramática, enquanto que as de estrutura são mais apropriadas a gramáticas, e fazem uma análise da estrutura em grafo da gramática. No ultimo ponto, apresentámos dois tipos de métricas que apresentam uma nova abordagem, uma baseada em tabelas LR e outra baseada na linguagem gerada. Não aprofundámos estas novas abordagens,apesar de serem abordagens novas, pois os resultados encontrados eram apenas experimentais e preferirmos dedicar a nossa abordagem a métricas mais estabelecidas.