Resumo do artigo Utilizando sintese de programas para analises de programas de Cristina David, Daniel Kroening and Matt Lewis

Fábio Moreira Duarte

Sumário

Resumo

No artigo define-se um fragmento da logica de segunda order com restrições quantificativas expressivas para capturar numerosos problemas de analise estatistica. Chamada de fragmento de sintese. Satisfabilidade da formula no fragmento de sintese é decidida sobre dominios finitos., o problema é NEXPTIME-complete. Se uma formula desse fragmento é satisfatorio, a solução consiste de uma tarefa satisfatoria de variaveis de segunda ordem para funções de dominio finito. Para encontrar tais soluções, sintetiza-se programas para computar as funções. Pode utilizar o sintetizador como procedimento de decisão para fragmento de sintese de logica de segunda order, permitindo o uso como backend para tarega de analise de programas.

1 Introdução

Programas de analise estatistica procuram por um prova do programa. A prova toma forma de invariantes do programa, para encontrar erros que é contra modelo, para analise de termino utiliza-se uma função de ranking, para não terminais utiliza-se conjunto de recorrencia. Encontrar cada prova sujeito-se a extensiva pesquisa resultando em varias tecnicas.

O processo de busca pela prova passa por laços de refinamento de duas fases: Uma fase heuristica, ajustando a profundidade de desenrolamento, refinando o conjunto de predicativos, selecionando um modelo, aplicando um operador de alargamento, a outra fase envolve chamar um programa de decisão. A heurística restringe o universo de possíveis provas a apenas um candidato, validado por outra fase. O desconhecido na primeira fase são as provas. O desconhecido da segunda fase são variaveis de programa. A primeira fase reduz um problema de segunda order a um de primeira ordem.

Dado o designe dificulta para separar a formulação de problemas para partes do processo de solução, resultando em analises confusas e frageis. Mudanças no processo de busca causa mudança em toda analise. Gostaria-se de um modelo onde a busca pela solução é encapsulada e separa completamente da formulação do problema.

O designe existente é dedicado pelo estado da arte em solucionar tecnologia. A tecnologia SAT/SMT permite solucionar problemas indutriais.

O artigo toma um passo endereçando tal situação, identificando um fragmento de logica de segunda ordem com quantificação restrita, expressiva para capturar problemas de analise estatistica e solucionando sintese de programa. Chamado de fragmento de sintese. O fragmento de sintese é decidido sobre dominios finitos e o problema de decisão é NEXPTIME-complete.

Se uma formula no fragmento de sintese é satisfazivel, a solução consiste da tarefa de variaveis de segunda ordem para funções sobre dominios finitos. Cada função é computada por um programa que possa ser sintetizado. Correspondência entre satisfação logica e sintese do programa permite desenvolver o processo de decisão para o fragmento de sintese como programa de sintese. Permitindo o uso como backend em tarefas de analise de programa.

O programa sintetizador usa combinações de controle modelos delimitados, controle de modelos de estado e programação genetica. Estrategia de generalização encontra soluções simples que solucionam um caso restrito do programa, logo tenta generaliza-lo para uma solução completa.

2 O fragmento de sintese

Definição 1 (Fragmento de sintese (SF)): Uma formula do fragmento de sintese é da forma:

Existe P1 ... Pm.Q1X1 ... QnXn. sigma(P1, ... , Pm, x1,...Xn)

Onde Pi é alcance das funções. Qi são existe e contem, O xi é o alcance de termo fundamentais e sigma é a formula livre de quantificador.

3 Especificações do programa de analise no fragmento de sintese

Problemas de analise pode ser reduzidos a problema de encontra solução para limitantes de segunda ordem.

Variantes de segurança. Dado uma afirmação de segurança A, uma variante de afirmação é o conjunto de estados S que respeitam a relação de transição do programa, excluem estados de erro.

Terminações e não terminações podem ser codificadas com a formula, onde W é uma variante indutiva do laço estabelecido pelo estado inicial I, se o laço G guardião e conheceu, e R é uma função de classificação restrita por W.

4 Fragmento de sintese sobre dominios finitos

Interceptando os termos fundamentais sobre dominio finito D, o fragmento de sintese é decidivel é o problema de decisão é NEXPTIME-complete. SFd denota o fragmento de sintese sobre dominio D.

Teorema 1(SFd é NEXPTIME-complete): Para uma instancia de definição 1 com n variaveis de primeira ordem, onde os termos fundamentais são interpretados sobre D, checar a veracidade da formula é NEXPTIME-complete.

Preopupa-se em construi um solucionador para SFd. Um modelo de segurança para a formula SFd satisfatorio é uma tarefa de mapeamento para cada variavel de segunda ordem para funções do tipo e aridade apropriadas. Quando decidindo uma instancia SFd em particar é satisfázivel, deve-se pensar como a solução e a função é codificada.

Irá gerar-se programas de estado finitos que computem as funções e representam tais programas como lista finita de instruções na forma SSA.

Teorema 2: Cada total, função finita é computada por ao menso um programa na linguagem.

Teorema 3: A representação é oticamente concisa.

Definição 2 (Formula de sintese finita) existe P. contem x pertence D sigma(P,x)

A especificação sigma, apresenta a função exec(P,x) que retorna o resultado o programa execução P com entrada x. P não pode conter laço ou recursão, exec é uma função total.

Teorema 4 (SFd é tempo polinomial redutivel para sintese finita): Cada instância da definição 1, onde os termos fundamentais são interpretado sobre D é de tempo polinomial redutivel.

Corolário 1: Programa de sintese com estados finitos é NEXPTIME-complete.

Esboça o design do processo de decisão para SFd: converge o problema de segurabilidade do SFd para um problema de sintese finita equalizado., solucionando com um programa de sintese com estados finitos.

5 Decidindo SFd via programa de sintese de estado finito

Apresenta algoritmos de sintese de estado finito utilizado para decidir a formula de segurabilidade do SFd. Descreve-se o processo de decisão geral, depois detalha como o processo de decisão é instanciado para o programa de sintese de estado finito.

5.1 Algoritmo de sintese de próposito geral

Algorithm 1 Abstract refinement algorithm

```
1: function SYNTH(inputs)
2:     (i_1, ..., i_N) ← inputs
3:     query ← ∃P.σ(i_1, P)∧...∧σ(i_N, P)
4:     result ← decide(query)
5:     if result.satisfiable then
6:         return result.model
7:     else
8:         return UNSAT

9: function VERIF(P)
10:    query ← ∃x.¬σ(x, P)
11:    result ← decide(query)
12:    if result.satisfiable then
13:        return result.model
14:    else
15:        return valid

16: function REFINEMENT LOOP
17:    inputs ← ∅
18:    loop
19:        candidate ← SYNTH(inputs)
20:        if candidate = UNSAT then
21:            return UNSAT
22:        res ← VERIF(candidate)
23:        if res = valid then
24:            return candidate
25:        else
26:            inputs ← inputs ∪ res
```

Utiliza-se sintese indutiva guiada de exemplo para encontrar um programa satisfátorio. Algoritmo 1 é divido em SYNTH e VERIF, interage com um conjunto finitos de vetores de entrada.

5.2 Sintese de estado finito

As características de programas C- são a segurabilidade pode ser decidida utilizando query para um modelo controlado. Um programa C- segue as seguintes restrições:

5.3

Laçõs devem ter uma constante de limite.

Toda recurão no programa deve ser limitada por uma constante de profundidade.

Todas lista devem ser estatisticamente alocada e de tamanho constante.

5.4 Estrátegias de geração de candidatos

Um candidato a solução P é escrito em linguagem RISC-like.

Para porções SYNTH de laçõs CEGIS, constroi-se um programa C- SYNTH.C que recebe os parametros candidatos P e entradas de teste. O programa contem afirmações quanto falha se P encontra a especificação para cada entrada. Encontrar novos programas candidatos é equivalente a checar a segurabilidade de SYNTH.C.

Aplica as estrategias em paralelo para encontrar candidatos.

5.5

Busca por prova explicita

Verificação simbolica do modelo vinculado

Programação genetica e evolução incremental

6 Procurando o espaço para possiveis soluções

Em algoritmos de sintese busca-se por programas candidatos a espaço. O componente chave é parametrizando a linguagem L. Começa sintetizando um programa no ponto mais baixo da estrutura e aumenta os parametros de L, ate alcansar um ponto de sintese de sucesso.

Dado um processo de busca automatizada, a parametrização aumenta a eficiência do sistema desde linguagem baixas da estrutura decidingo segurança.

6.1 Parametros da linguagem L