Лабораторная работа 1

Посредством генерации подходящей SMT-модели:

1. (вар. 2, 4, 5) Реализовать проверку завершаемости TRS на линейных интерполяциях функций.
2. (вар. 6, 7, 8) Реализовать проверку завершаемости SRS на линейных ординальных функциях до ω^2.
3. (вар. 0, 1, 3, 9) Реализовать проверку завершаемости SRS посредством построения оценки линейных операторах размерности 2 над арктическим полукольцом.

Подразумевается, что входными данными является произвольная система переписывания термов, а не заранее известная.

Синтаксис входных данных

Здесь красным выделены элементы входного языка, чёрным — элементы метаязыка.

Синтаксис записи входных данных для TRS (задача 1):

$$
\begin{array}{rcl}
\textcolor{red}{\text{variables}} & \textcolor{red}{=} & ([\text{буква}]\textcolor{red}{,})^{*}\ [\text{буква}] \\
([\text{терм}] & \textcolor{red}{=} & [\text{терм}])^{+}
\end{array}
$$

$$
\begin{array}{rcl}
[\text{терм}] & ::= & [\text{переменная}] \\
 & & \mid [\text{конструктор}]\ (\textcolor{red}{(}([\text{терм}]\textcolor{red}{,})^{*}[\text{терм}]\textcolor{red}{)})? \\
[\text{переменная}] & ::= & [\text{буква}] \\
[\text{конструктор}] & ::= & [\text{буква}]
\end{array}
$$

Множества имён переменных и конструкторов считаем непересекающимися. Арность конструкторов полагаем равной либо 0, либо 1, либо 2.

Синтаксис входных данных

Здесь красным выделены элементы входного языка, чёрным — элементы метаязыка.

Синтаксис записи входных данных для SRS (задача 2,3):

([строка] -> [строка])$^+$

То есть все буквы по умолчанию считаются конструкторами местности 1.

Постановка задачи вар.1

- Всем конструкторам следует придать интерпретации в виде линейных функций арифметики. Вычислить необходимые композиции функций, интерпретирующие левые и правые части правил переписывания, и построить неравенства по каждому правилу отдельно по каждой переменной.
- Далее требуется породить файл smt2-спецификации для решателя в логике QF_NIA, описывающий условия завершаемости и задающий ограничения на параметры.
- Отдельный скрипт должен вызывать всё вместе: генератор и smt-солвер, после чего возвращать ответ: интерполяцию, доказывающую завершаемость системы, либо сообщение, что интерполяция не была найдена.

Фундированность

Определение

Частичный порядок $\preceq$ является фундированным (wfo) на множестве M, если в M не существует бесконечных нисходящих цепочек относительно $\preceq$ (иногда используют термин анти-нётеровый, артиновый, или просто нётеровый).

Частичный порядок $\preceq$ является монотонным в алгебре A, если $\forall f, t_1, ..., t_n, s, s'(s \preceq s' \Rightarrow f(t_1, \ldots, s, \ldots, t_n) \preceq f(t_1, \ldots, s', \ldots, t_n))$ (строго монотонным, если при этом неверно обратное).

Завершаемость

Определение

Фундированная монотонная алгебра (ФуМА) над множеством функциональных символов F — это фундированное множество $\langle A, > \rangle$ такое, что для каждого функционального символа $f \in F$ существует функция $f_A : A^n \to A$, строго монотонная по каждому из аргументов.

Определим расширение произвольного отображения σ из множества переменных в A следующим образом:

- $[x, \sigma] = \sigma(x)$;
- $[f(t_1, \ldots, t_n), \sigma] = f_A([t_1, \sigma], \ldots, [t_n, \sigma])$.

Завершаемость

Совместность

TRS $\{l_i \to r_i\}$ совместна с ФуМА $A \Leftrightarrow$ для всех i и для всех σ выполняется условие $[l_i, \sigma] > [r_i, \sigma]$.

Теорема

TRS не порождает бесконечных вычислений (завершается), если и только если существует совместная с ней ФуМА.

ФуМА, совместные с TRS

Стандартные способы определения f_A:

- лексикографический порядок на множестве имён F + отношение подтерма;
- построение монотонно возрастающей (по каждому аргументу) числовой функции, соответствующей f_A.

Оба случая подразумевают, что в построенной модели целое больше части, т.е. всегда выполняется $f(t) > t$.

Пример

Проверить завершаемость TRS методом построения монотонной функции:

$$f(g(x, y)) \to g(h(y), x)$$
$$h(f(x)) \to f(x).$$

- Завершаемость по второму правилу переписывания автоматически выполняется по свойству подтерма. Поэтому то, что функция f стоит на двух его сторонах, не дает никаких указаний относительно того, стоит ли делать f_A быстро растущей или медленно. Все подсказки содержатся только в первом правиле переписывания.
- По первому правилу переписывания видно, что f_A надо делать большой (f стоит только слева), а h нет (h есть только справа). Положим $f_A(x) = 10 \cdot (x + 1)$, $h_A(x) = x + 1$. Тогда должно выполняться $10 \cdot (g_A(x, y) + 1) > g(y + 1, x)$. Этому неравенству удовлетворяет, например, $g_A(x, y) = x + y + 1$.

Общие комментарии

- Не обязательно добиваться выполнения неравенства на образах f_A на всём множестве $\mathbb{N}$. Поскольку любой отрезок $\mathbb{N}$ от k и до бесконечности фундирован, а все образы f_A монотонны, они замкнуты на этом отрезке. Поэтому, если неравенство не выполняется для нескольких первых чисел натурального ряда, этим можно пренебречь.
- Таким образом, формализация условия строгой монотонности в модели может быть переформулирована как «строгое возрастание по коэффициентам при каждой переменной и нестрогое — по свободному члену, или строгое возрастание по свободному члену и нестрогое — по коэффициентам по переменным».

Построение системы неравенств

Переформулируем задачу оценки построения линейных интерпретаций f_A, h_A, g_A в параметрах:

$$f_A(x) = a_1 \cdot x + a_2$$
$$g_A(x, y) = b_1 \cdot x + b_2 \cdot y + b_3$$
$$h_A(x) = c_1 \cdot x + c_2$$

Построим функции для левой и правой части правила $f(g(x, y)) \to g(h(y), x)$:

$$f_A(g_A(x, y)) = a_1 \cdot (b_1 \cdot x + b_2 \cdot y + b_3) + a_2$$
$$g_A(h_A(y), x) = b_1 \cdot (c_1 \cdot y + c_2) + b_2 \cdot x + b_3$$

Построение системы неравенств

Построим функции для левой и правой части правила $f(g(x, y)) \to g(h(y), x)$:

$$f_A(g_A(x, y)) = a_1 \cdot (b_1 \cdot x + b_2 \cdot y + b_3) + a_2$$
$$g_A(h_A(y), x) = b_1 \cdot (c_1 \cdot y + c_2) + b_2 \cdot x + b_3$$

Тогда соответствующие системы неравенств будут выглядеть следующим образом:

$$\begin{cases} a_1 \cdot b_1 \geqslant b_2 \\ a_1 \cdot b_2 \geqslant b_1 \cdot c_1 \\ a_1 \cdot b_3 + a_2 > b_1 \cdot c_2 + b_3 \end{cases} \qquad \begin{cases} a_1 \cdot b_1 > b_2 \\ a_1 \cdot b_2 > b_1 \cdot c_1 \\ a_1 \cdot b_3 + a_2 \geqslant b_1 \cdot c_2 + b_3 \end{cases}$$

Нахождение значений, подходящих хотя бы под одну, обосновывает завершаемость правила. Однако нужно не забыть наложить подходящие ограничения на константы a_i, b_i, c_i, гарантирующие строгую монотонность f_A, g_A, h_A.

Ординалы

Определение

Рассмотрим множество M с определенным на нем полным (линейным, фундированным) порядком $<$. Ординал τ — это порядок множества $\langle M, < \rangle$ (иногда в виде τ рассматривается само это множество). Если существует биекция f из $\langle M, < \rangle$ в $\langle M', <' \rangle$, являющаяся гомоморфизмом, то M и M' имеют одинаковые порядки.

Ординал любого множества $\{1, 2, \ldots, k\}$, где $k \in \mathbb{N}$ — это k. Ординал $\mathbb{N}$ — это ω.

Предельные ординалы

Определение

Ординал τ предельный, если не существует τ_0 такого, что $\tau = \tau_0 + 1$.

Существование предельных ординалов делает ординальную арифметику некоммутативной, поскольку правый элемент сложения и умножения может поглощать левый.

Ординальная арифметика

- Сложение:
  $\alpha + 0 = \alpha$;
  $\alpha + (\beta + 1) = (\alpha + \beta) + 1$;
  β — предельный $\Rightarrow \alpha + \beta = \lim_{\gamma < \beta}(\alpha + \gamma)$.
- Умножение:
  $\alpha \cdot 0 = 0$;
  $\alpha \cdot (\beta + 1) = (\alpha \cdot \beta) + \alpha$;
  β — предельный $\Rightarrow \alpha \cdot \beta = \lim_{\gamma < \beta}(\alpha \cdot \gamma)$.

Дистрибутивность только левая: $\alpha \cdot (\beta + \gamma) = \alpha \cdot \beta + \alpha \cdot \gamma$.

Примеры ординальных вычислений

- Вычислим $(\omega + 1) + \omega \cdot 2$. Умножение можно раскрыть стандартным образом, поскольку 2 — не предельный ординал. Получается $\omega + 1 + \omega + \omega$. Чтобы вычислить $1 + \omega$, перейдём к пределу: $\lim_{\gamma < \omega}(1 + \gamma)$. Искомый предел есть ω, и результат — $\omega + \omega + \omega$, который можно свернуть в $\omega \cdot 3$ по правилу умножения.
- Вычислим $\omega \cdot \omega^{\omega} + \omega^{\omega^{\omega}}$. Здесь все ординалы предельные, поэтому сразу же строим предел для подтерма $\omega \cdot \omega^{\omega}$: $\lim_{\gamma < \omega^{\omega}}(\omega \cdot \gamma)$. Поскольку γ включает ряд ω^{k}, предел будет равен ω^{ω}. Осталось вычислить $\lim_{\gamma < \omega^{\omega^{\omega}}}(\omega^{\omega} + \gamma)$. По аналогичным соображениям получается ординал $\omega^{\omega^{\omega}}$.

Данные вычисления приведены для ознакомления. В решении задачи можно будет обойтись базовыми преобразованиями на основе дистрибутивности, без раскрытия пределов.

Вычисления на малых ординалах

Умножение ординала до ω^ω на ординал до ω^2:

$$\left(\omega^n \cdot a_n + \omega^{n-1} \cdot a_{n-1} + \cdots + \omega \cdot a_1 + a_0\right) \cdot (\omega \cdot b_1 + b_0)$$
$$= \omega^{n+1} \cdot b_1 + \omega^n \cdot (a_n b_0) + \omega^{n-1} \cdot a_{n-1} + \ldots \omega \cdot a_1 + a_0$$

Ординал $w^n \cdot a_n + \cdots + a_0$ больше, чем $w^m \cdot b_m + \ldots b_0$, где $a_n \neq 0$ и $m \neq 0$, если:

- или $n > m$;
- или $n = m$ и $\exists i(a_i > b_i \,\&\, \forall j(j > i \Rightarrow a_j = b_j))$.

Таким образом, ординалы до ω^ω сравниваются лексикографически относительно степеней от старшей к младшей.

Постановка задачи вар.2

- Всем конструкторам следует придать интерпретации в виде линейных функций с ординальными коэффициентами, меньшими ω^2 (то есть имеющими вид $\omega \cdot \alpha_1 + \alpha_2$). В этом варианте работа с SRS, значит, все интерполяции имеют вид $F_{\mathbb{O}}(x) = (\omega \cdot a_F + b_F) \cdot x + \omega \cdot c_F + d_F$. Построить неравенства на возрастание коэффициента при x и на возрастание свободного коэффициента.
- Далее требуется породить файл smt2-спецификации для решателя в выбранной вами логике, описывающий условия завершаемости.
- Отдельный скрипт вызывает всё вместе: генератор и smt-солвер, после чего возвращает ответ: интерполяцию, доказывающую завершаемость системы, либо сообщение, что интерполяция не была найдена.

Арктическое полукольцо

Полукольцо над носителем $\{-\infty\} \cup \mathbb{N}$, где $\oplus$ — это операция максимума, а $\otimes$ — операция сложения, называется арктическим.

Положим $> = \gg$, где $\gg = \{(x, y) \mid x > y \vee (x = y = -\infty)\}$.
Проблема: $\gg$ не фундирован. Поэтому необходимо наложить дополнительные условия на входные данные, чтобы вынудить его быть фундированным.

Арктическая завершаемость

Скажем, что матрица (вектор) конечна, если её элемент с индексами 1,1 (индексом 1) не равен $-\infty$.

Если интерпретация функций TRS $\mathcal{R}$ линейными операторами в арктическом полукольце такова, что:

- все конструкторы имеют местность 0 или 1;
- для всех констант их интерпретация конечна;
- для всех унарных функций их интерпретация конечна;
- $[l] \gg [r]$ для всех правил из $\mathcal{R}$;

тогда вычисление $\mathcal{R}$ всегда завершается.

Формализация

- Соответствующие функции имеют вид
$$F_{\mathbb{A}}(x) = \begin{pmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{pmatrix} \otimes x \oplus \begin{pmatrix} b_1 \\ b_2 \end{pmatrix},$$ при $x \in (\mathbb{N} \cup \{-\infty\})^2$.
При этом дополнительно требуем $a_{11} \neq -\infty$, и либо $b_1 \neq -\infty$, либо отсутствие свободного коэффициента (то есть интерпретацию вида $F_{\mathbb{A}} = \begin{pmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{pmatrix} \otimes x$).
- Порядок $\gg$ распространяется на матрицы и вектора покомпонентно.

Для неискушённых

Постановка задачи подразумевает, что строятся деревья композиции арифметических действий над параметрами, которые затем транслируются в smt2-язык в терминах этих же параметров. Например, в случае простейшей линейной интерполяции действие выглядит примерно так:

Исходное правило

$$f(g(x)) = g(f(x))$$

Интерпретации

$$\begin{cases} f(x) = a_1 \cdot x + a_0 \\ g(x) = b_1 \cdot x + b_0 \end{cases}$$

Композиции

$$\begin{cases} f(g(x)) = a_1 \cdot (b_1 \cdot x + b_0) + a_0 \\ g(f(x)) = b_1 \cdot (a_1 \cdot x + a_0) + b_0 \end{cases}$$

Для неискушённых

Исходное правило

$$f(g(x)) = g(f(x))$$

Интерпретации

$$\begin{cases} f(x) = a_1 \cdot x + a_0 \\ g(x) = b_1 \cdot x + b_0 \end{cases}$$

Композиции

$$\begin{cases} f(g(x)) = a_1 \cdot (b_1 \cdot x + b_0) + a_0 \\ g(f(x)) = b_1 \cdot (a_1 \cdot x + a_0) + b_0 \end{cases}$$

Транслируемая система неравенств

$a_1 \cdot b_1 \geqslant b_1 \cdot a_1$ (по коэффициенту x)

$a_1 \cdot b_0 + a_0 \geqslant b_1 \cdot a_0 + b_0$ (по свободному коэффициенту)

$(a_1 \cdot b_1 > b_1 \cdot a_1) \vee (a_1 \cdot b_0 + a_0 > b_1 \cdot a_0 + b_0)$

(условие строгости убывания)

$a_1 \geqslant 1 \;\&\; b_1 \geqslant 1 \;\&\; a_0 \geqslant 0 \;\&\; b_0 \geqslant 0$

(нестрогая монотонность f, g)

$((a_1 > 1) \vee (a_0 > 0)) \;\&\; ((b_1 > 1) \vee (b_0 > 0))$

(строгая монотонность f, g)

Синтаксис SMT2

- При решении лабораторных работ рекомендуется использовать теорию нелинейной целочисленной арифметики: `(set-logic QF_NIA)` (директиву обычно включают в самое начало smt2-файла).
- Объявление параметра: `(declare-fun X () [Тип])`, где тип может быть `Int` или `Bool` (для `NIA`).
- Объявление аксиомы: `(assert [выражение])`, где выражение — предикат или логическая формула над предикатами.
- Доступны предикаты сравнения (=, >=, <=, >, <), логические операции: (`or`, `not`, `and`, `=>`); а также оператор if-then-else `ite`. Например, `(ite (< x y) x y)` возвращает минимум из двух значений.
- Построение новых функций: `define-fun`. Например, так можно объявить функцию возведения в квадрат на $\mathbb{R}$: `(define-fun sq ((a Real)) Real (* a a))`
- Проверка модели на непустоту — директива `(check-sat)`.
- Чтобы находились значения параметров, удовлетворяющие модели, требуется включить директиву `(get-model)`.