Реализация дистиллятора для простого функционального языка на Haskell

Автор: Винник Екатерина Петровна
Научный руководитель: доцент кафедры информатики,
к. ф.-м.н. С. В. Григорьев
Консультант: доцент кафедры системного программирования,
к. ф.-м.н. Д. А. Березун
Рецензент: программист ООО «Интеллиджей Лабс» Е. А. Вербицкая

Санкт-Петербургский государственный университет
Системное программирование

24.05.2022

Актуальность

- Создание промежуточных структур данных ухудшает производительность приложений
- Существуют решения для частных случаев
  - STREAM FUSION
  - XLA

Актуальность

- Создание промежуточных структур данных ухудшает производительность приложений
- Существуют решения для частных случаев
  - STREAM FUSION
  - XLA
- Общего практического решения не предложено

Актуальность

- Создание промежуточных структур данных ухудшает производительность приложений
- Существуют решения для частных случаев
  - STREAM FUSION
  - XLA
- Общего практического решения не предложено
- Академические исследования
  - Дефорестация
  - Частичные вычисления
  - Суперкомпиляция
  - **Дистилляция**

Дистилляция

- Основана на символьном исполнении кода

Дистилляция

- Основана на символьном исполнении кода
- ✚ Позволяет достичь суперлинейного ускорения
- ✚ Позволяет трансформировать большее множество программ

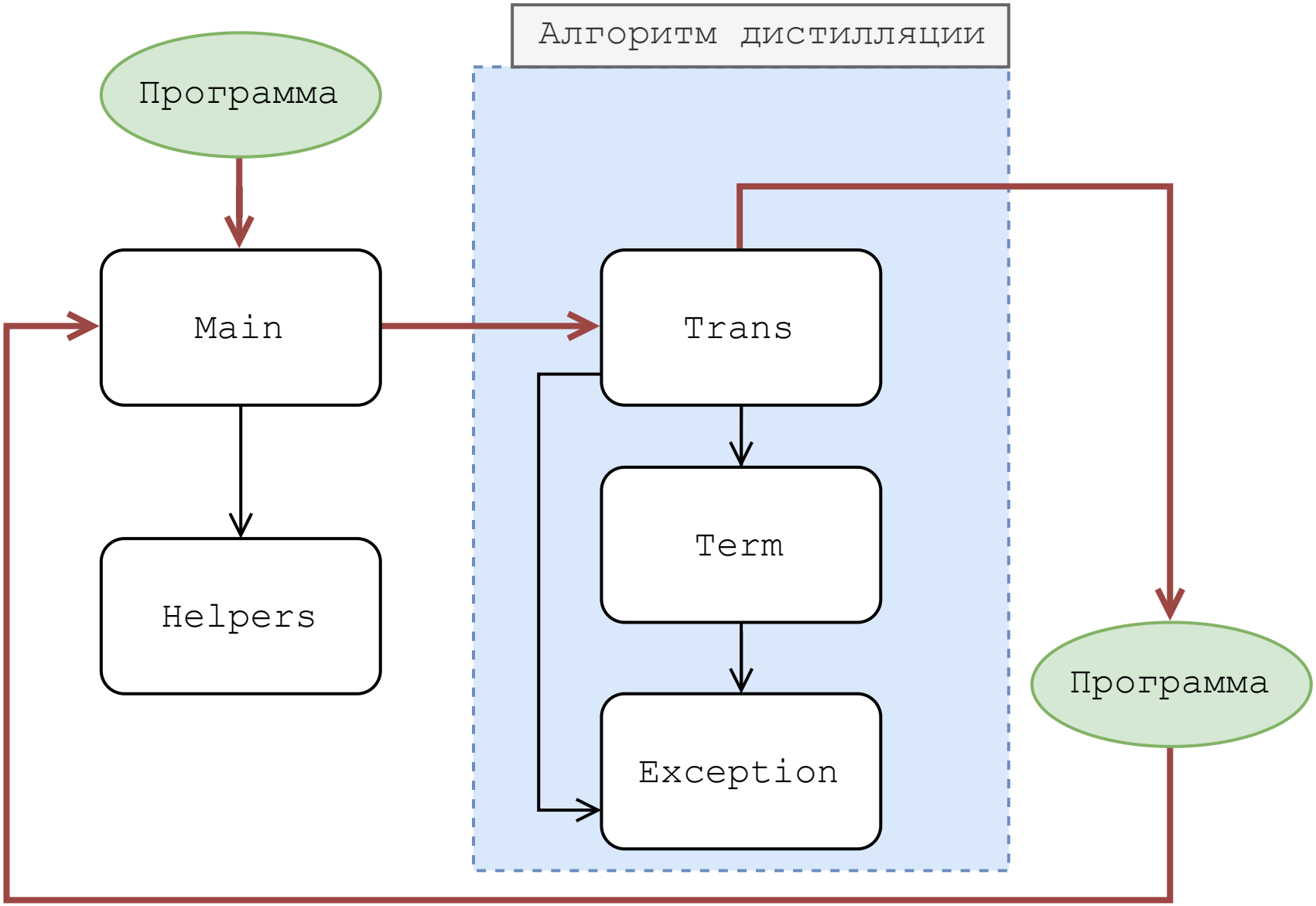
Дистилляция

- Основана на символьном исполнении кода

+ Позволяет достичь суперлинейного ускорения
+ Позволяет трансформировать большее множество программ

− Существует **единственная** реализация
− Код не согласован со статьями
− Плохая архитектура
  - Модули ответственны за большое количество этапов алгоритма
  - Структура используемых типов усложняет изучение и модификацию реализации
  - Отсутствие тестового покрытия

Существующая архитектура



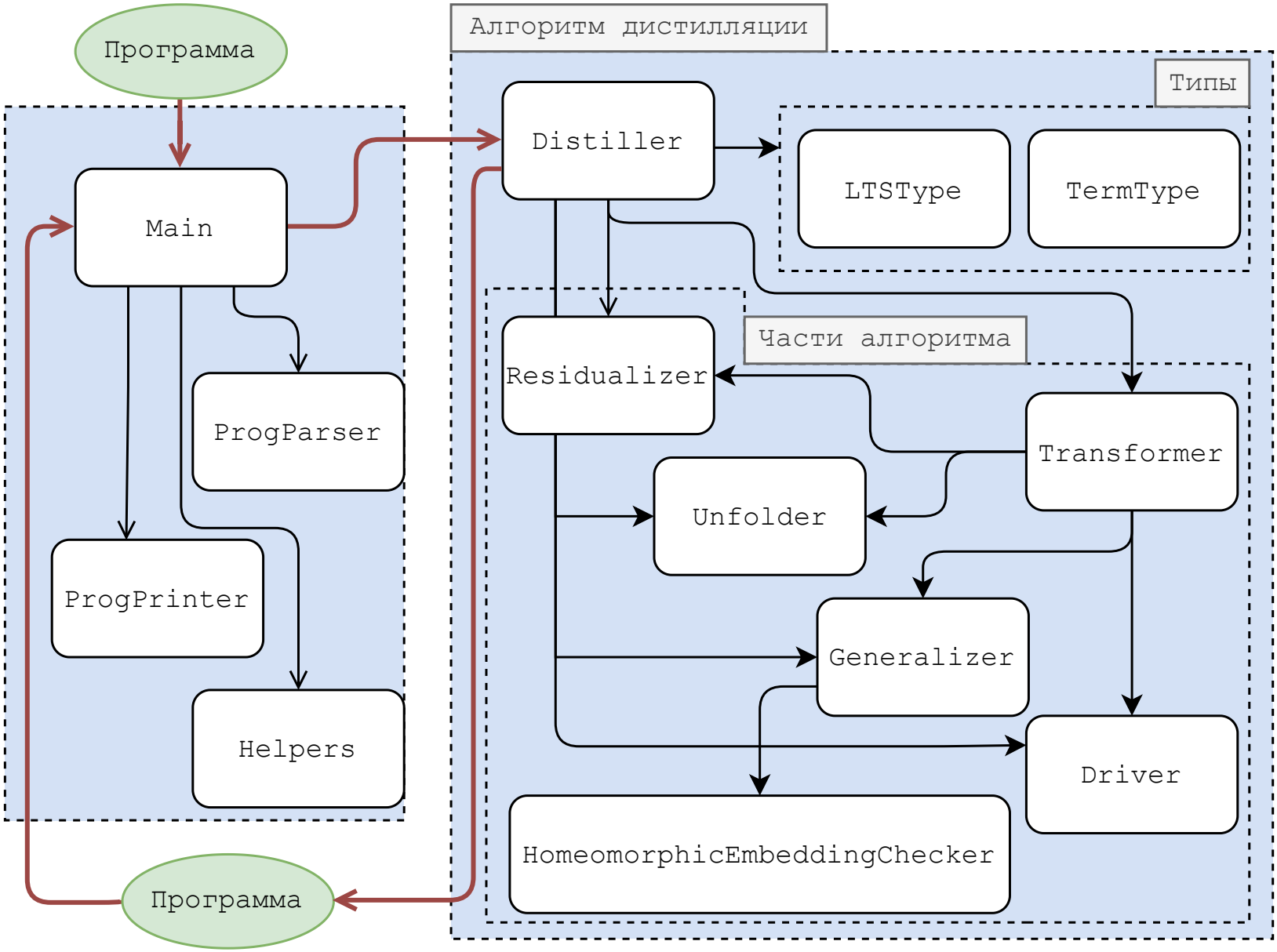
Постановка задачи

Цель: реализация упрощенного алгоритма дистилляции в существующем проекте DISTILLER

Задачи:

- Перепроектировать существующее решение
- Реализовать упрощенную версию алгоритма дистилляции
- Подготовить тестовую инфраструктуру
- Провести тестирование дистиллятора с помощью различных методик

Разработанная архитектура дистиллятора



Особенности реализации

- Язык реализации – Haskell
- Интерфейс пользователя отделен от реализации
- Разработаны новые типы, упрощающие чтение и реализацию алгоритма
- Подготовлена инфраструктура для тестирования проекта
  - Подключена тестовая платформа Tasty
  - Добавлена непрерывная интеграция с использованием GitHub Actions

Тестирование

- Применены различные методы тестирования:
  - Функциональное тестирование
  - Интеграционное тестирование
  - Модульное тестирование
  - Тестирование на основе свойств программ
- Реализовано более 120 тестов
- Во время тестирования были найдены неточности в описании алгоритма, приводящие к проблемам реализации:
  - Проблема неконсистентного введения новых функций во время дистилляции
  - Проблема распространения информации в алгоритме

Достигнутые результаты

- Перепроектирована архитектура существующего решения
- Реализована и интегрирована упрощенная версия алгоритма дистилляции
- Подготовлена инфраструктура для тестирования:
  - Добавлена непрерывная интеграция с помощью GitHub Actions
  - Проведено тестирование различными методами с использованием Tasty
- Во время тестирования были найдены неточности в описании алгоритма, приводящие к проблемам реализации:
  - Проблема неконсистентного введения новых функций
  - Проблема распространения информации в алгоритме

  Проблемы для полученной реализации были решены.
- Приняты к докладу на Verification Program Transformation workshop 2022

Пояснение к проблеме распространения информации в алгоритме

1. Алгоритм дистилляции имеет множество этапов
2. При вычислении алгоритма используется множество параметров, часть из которых передается между этапами, а часть — нет
3. Выстраиваемые в процессе работы алгоритма конструкции должны быть доступны во всех ветках графа программы

Трансформер 0 уровня

$$
\begin{aligned}
&\mathcal{T}_0[\![x]\!]\ \kappa\ \rho\ \theta\ \Delta \quad = \mathcal{T}_0'[\![x \to (x, \mathbf{0})]\!]\ \kappa\ \rho\ \theta\ \Delta \\
&\mathcal{T}_0[\![e = c\ e_1 \ldots e_n]\!]\ \langle\rangle\ \rho\ \theta\ \Delta \\
&\qquad\qquad = e \to (c, \mathbf{0}), (\#1, \mathcal{T}_0[\![e_1]\!]\ \langle\rangle\ \rho\ \theta\ \Delta), \ldots, (\#n, \mathcal{T}_0[\![e_n]\!]\ \langle\rangle\ \rho\ \theta\ \Delta) \\
&\mathcal{T}_0[\![e = c\ e_1 \ldots e_n]\!]\ (\kappa = \langle(\mathbf{case}\ \bullet\ \mathbf{of}\ p_1 \Rightarrow e_1' \mid \cdots \mid p_k \Rightarrow e_k') : \kappa'\rangle)\ \rho\ \theta\ \Delta \\
&\qquad\qquad = (\kappa \bullet e) \to (\tau_c, \mathcal{T}_0[\![e_i'\{x_1 \mapsto e_1, \ldots, x_n \mapsto e_n\}]\!]\ \kappa'\ \rho\ \theta\ \Delta) \\
&\qquad\qquad \text{where } p_i = c\ x_1 \ldots x_n \\
&\mathcal{T}_0[\![e = \lambda x.e_0]\!]\ \langle\rangle\ \rho\ \theta\ \Delta \\
&\qquad\qquad = e \to (\lambda x, \mathcal{T}_0[\![e_0]\!]\ \langle\rangle\ \rho\ \theta\ \Delta) \\
&\mathcal{T}_0[\![e = \lambda x.e_0]\!]\ (\kappa = \langle(\bullet\ e_1) : \kappa'\rangle)\ \rho\ \theta\ \Delta \\
&\qquad\qquad = (\kappa \bullet e) \to (\tau_\beta, \mathcal{T}_0[\![e_0\{x \mapsto e_1\}]\!]\ \kappa'\ \rho\ \theta\ \Delta) \\
&\mathcal{T}_0[\![f]\!]\ \kappa\ \rho\ \theta\ \Delta \quad = \begin{cases} (\kappa \bullet f) \to (\tau_f, \mathbf{0}), & \text{if } \exists t' \in \rho, \sigma \bullet t' \approx_\sigma^\emptyset t \\ \mathcal{T}_0[\![\mathcal{R}[\![\mathcal{G}[\![t]\!][\![t']\!]\ \theta]\!]]\!]\ \langle\rangle\ \rho\ \theta\ \emptyset, & \text{if } \exists t' \in \rho, \sigma \bullet t' \bowtie_\sigma^\emptyset t \\ (\kappa \bullet f) \to (\tau_f, \mathcal{T}_0[\![\mathcal{U}[\![\kappa \bullet f]\!]\ \Delta]\!]\ \langle\rangle\ (\rho \cup \{t\})\ \theta\ \Delta), & \text{otherwise} \end{cases} \\
&\qquad\qquad \text{where } t = \mathcal{L}[\![\kappa \bullet f]\!]\ \emptyset\ \Delta \\
&\mathcal{T}_0[\![e_0\ e_1]\!]\ \kappa\ \rho\ \theta\ \Delta = \mathcal{T}_0[\![e_0]\!]\ \langle(\bullet\ e_1) : \kappa\rangle\ \rho\ \theta\ \Delta \\
&\mathcal{T}_0[\![\mathbf{case}\ e_0\ \mathbf{of}\ p_1 \Rightarrow e_1 \mid \cdots \mid p_k \Rightarrow e_k]\!]\ \kappa\ \rho\ \theta\ \Delta \\
&\qquad\qquad = \mathcal{T}_0[\![e_0]\!]\ \langle(\mathbf{case}\ \bullet\ \mathbf{of}\ p_1 \Rightarrow e_1 \mid \cdots \mid p_k \Rightarrow e_k) : \kappa\rangle\ \rho\ \theta\ \Delta \\
&\mathcal{T}_0[\![e = \mathbf{let}\ x = e_0\ \mathbf{in}\ e_1]\!]\ \kappa\ \rho\ \theta\ \Delta \\
&\qquad\qquad = (\kappa \bullet e) \to (\mathbf{let}, \mathcal{T}_0[\![e_1]\!]\ \kappa\ (\rho \cup \{x \mapsto e_0\})\ \theta\ \Delta), (x, \mathcal{T}_0[\![e_0]\!]\ \langle\rangle\ \rho\ \theta\ \Delta) \\
&\mathcal{T}_0[\![e_0\ \mathbf{where}\ f_1 = e_1 \ldots f_n = e_n]\!]\ \kappa\ \rho\ \theta\ \Delta \\
&\qquad\qquad = \mathcal{T}_0[\![e_0]\!]\ \kappa\ \rho\ \theta\ (\Delta \cup \{f_1 = e_1, \ldots, f_n = e_n\})
\end{aligned}
$$

Трансформер 0 уровня

$$\mathcal{T}_0'[\![t]\!]\ \langle\rangle\ \rho\ \theta\ \Delta = t$$

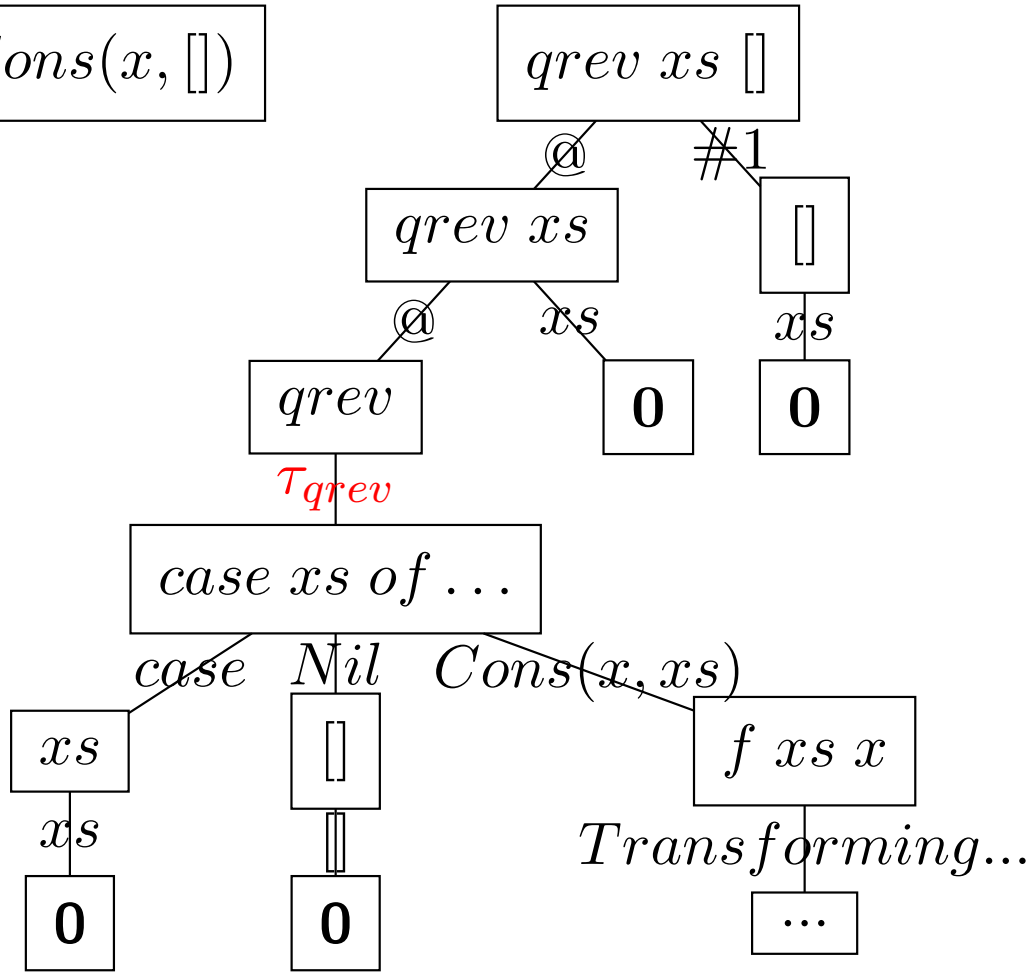
$$\mathcal{T}_0'[\![t]\!]\ \langle(\bullet\ e) : \kappa\rangle\ \rho\ \theta\ \Delta$$
$$= \mathcal{T}_0'[\![(t\ e) \to (@, t), (\#1, \mathcal{T}_0[\![e]\!]\ \langle\rangle\ \rho\ \theta\ \Delta)]\!]\ \kappa\ \rho\ \theta\ \Delta$$

$$\mathcal{T}_0'[\![x \to (x, \mathbf{0})]\!]\ \langle(\mathbf{case}\ \bullet\ \mathbf{of}\ p_1 \Rightarrow e_1' \mid \cdots \mid p_k \Rightarrow e_k') : \kappa\rangle\ \rho\ \theta\ \Delta$$
$$= (\mathbf{case}\ x\ \mathbf{of}\ p_1 \Rightarrow e_1' \mid \cdots \mid p_k \Rightarrow e_k') \to$$
$$(\mathbf{case}, t), (p_1, \mathcal{T}_0[\![(\kappa \bullet e_1')\{x \mapsto p_1\}]\!]\ \langle\rangle\ \rho\ \theta\ \Delta), \ldots, (p_k, \mathcal{T}_0[\![(\kappa \bullet e_k')\{x \mapsto p_k\}]\!]\ \langle\rangle\ \rho\ \theta\ \Delta)$$

$$\mathcal{T}_0'[\![t]\!]\ \langle(\mathbf{case}\ \bullet\ \mathbf{of}\ p_1 \Rightarrow e_1' \mid \cdots \mid p_k \Rightarrow e_k') : \kappa\rangle\ \rho\ \theta\ \Delta$$
$$= (\mathbf{case}\ (root(t))\ \mathbf{of}\ p_1 \Rightarrow e_1' \mid \cdots \mid p_k \Rightarrow e_k') \to$$
$$(\mathbf{case}, t), (p_1, \mathcal{T}_0[\![e_1']\!]\ \kappa\ \rho\ \theta\ ), \ldots, (p_k, \mathcal{T}_0[\![e_k']\!]\ \kappa\ \rho\ \theta\ \Delta)$$

Пояснение к проблеме неконсистентного введения новых функций

$$f\ xs\ x = qrev\ xs\ Cons(x, [])$$



Пример промежуточных структур данных

- `Array.map g (Array.map f arr)`
- `Array.map (g . f) arr`
- `Array.map g (Array.filter f arr)` — ?

Правила гомеоморфного вложения

$$t \lesssim^{\rho}_{\sigma} t', \quad \text{if } (t \lhd^{\rho}_{\sigma} t') \vee (t \bowtie^{\rho}_{\sigma} t')$$

$$t \lhd^{\rho}_{\sigma} (e \to (\alpha_1, t_1), \ldots, (\alpha_n, t_n)), \quad \text{if } \exists i \in \{1 \ldots n\} \bullet t \lesssim^{\rho}_{\sigma} t_i$$

$$(x \to (x, \mathbf{0})) \bowtie^{\rho}_{\sigma} (x' \to (x', \mathbf{0})), \quad \text{if } x\sigma = x'$$

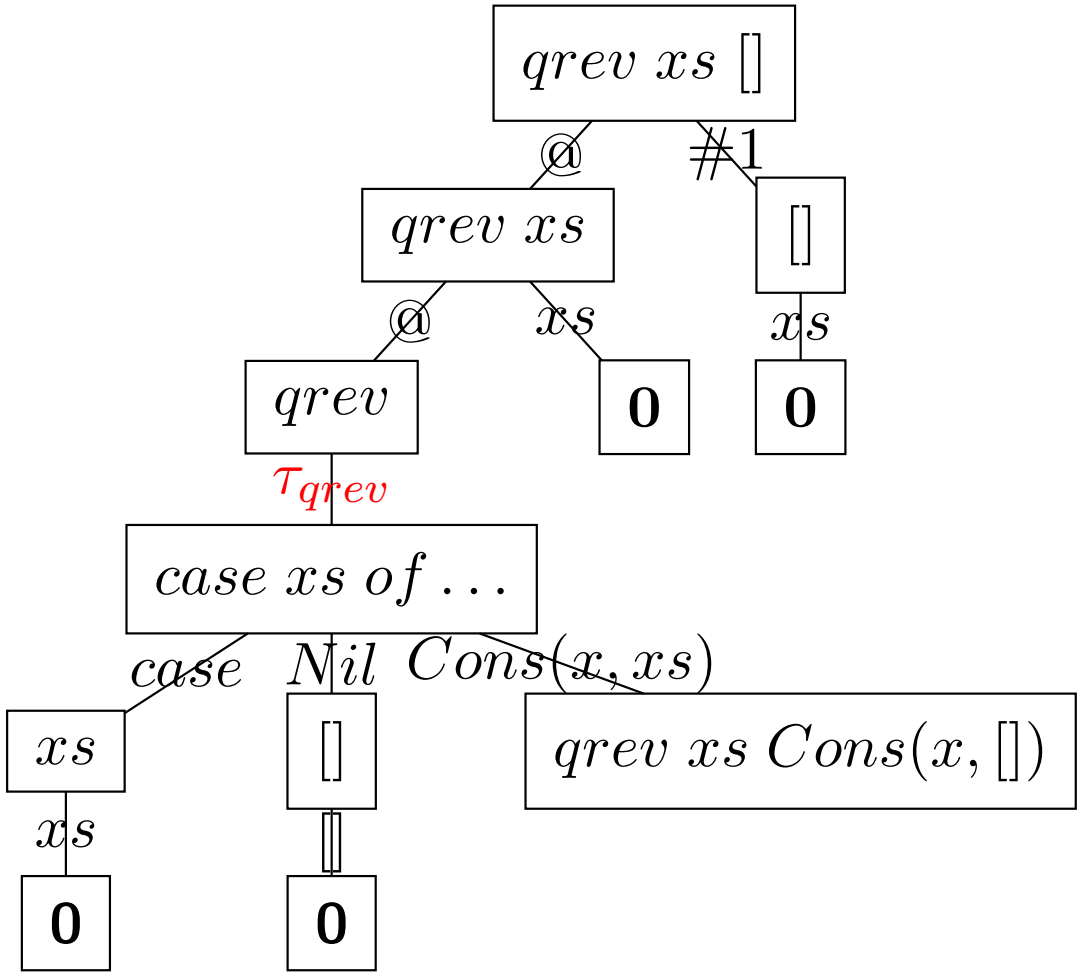
$$(e \to (\tau_f, t)) \bowtie^{\rho}_{\sigma} (e' \to (\tau_{f'}, t')), \quad \text{if } ((f, f') \in \rho) \vee (t \lesssim^{\rho \cup \{(f,f')\}}_{\sigma} t')$$

$$(e \to (\alpha_1, t_1), \ldots, (\alpha_n, t_n)) \bowtie^{\rho}_{\sigma} (e' \to (\alpha'_1, t'_1), \ldots, (\alpha'_n, t'_n)),$$
$$\text{if } \forall i \in \{1 \ldots n\} \bullet (\exists \sigma' \bullet (\alpha_i\ \sigma' = \alpha'_i \wedge t_i \lesssim^{\rho}_{\sigma \cup \sigma'} t'_i))$$

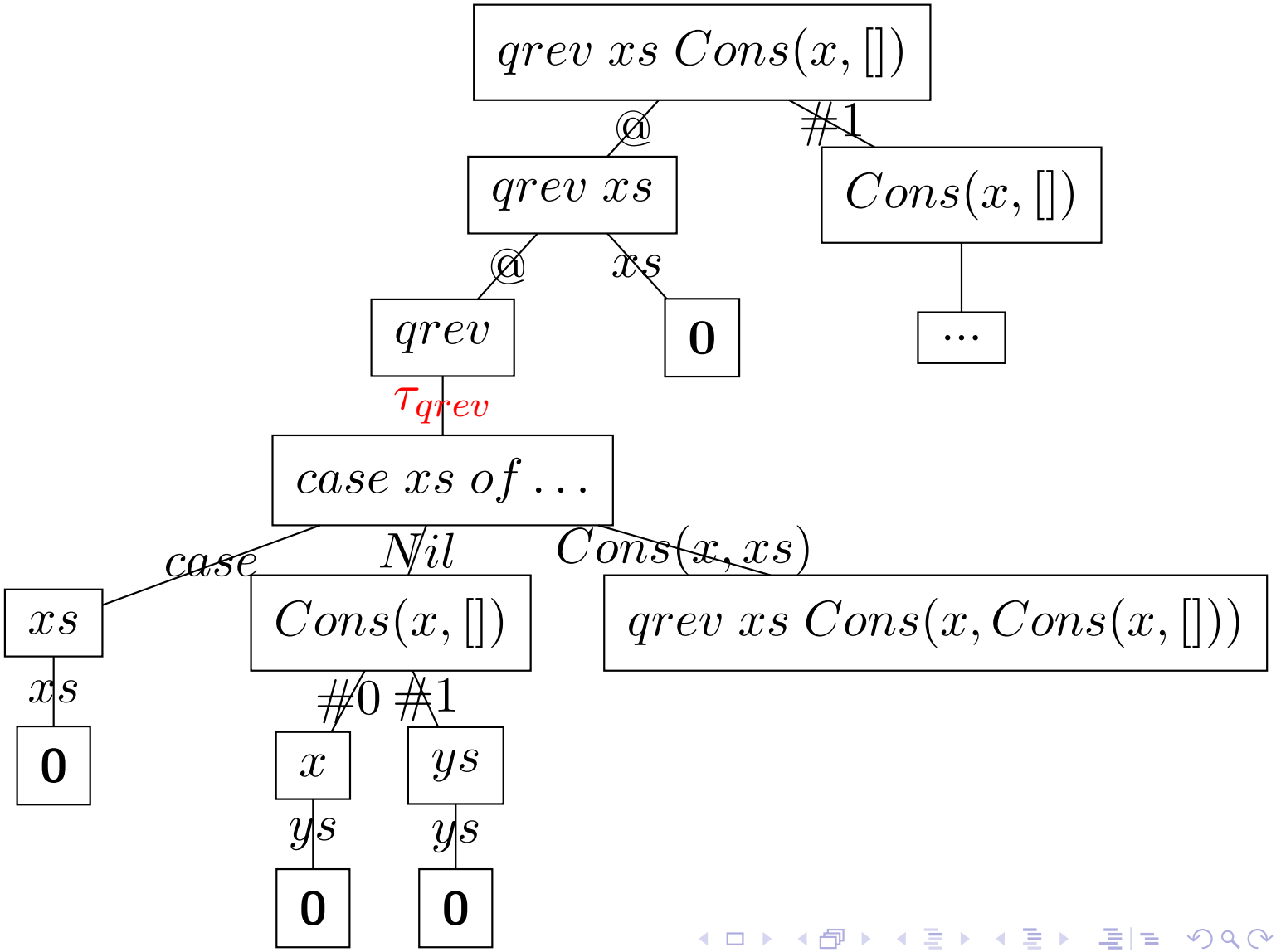
Техника дистилляции: построение помеченной системы переходов по программе

```
main = qrev xs []
qrev xs ys = case xs of
                 Nil         -> ys
               | Cons(x,xs) -> qrev xs Cons(x,ys);
```

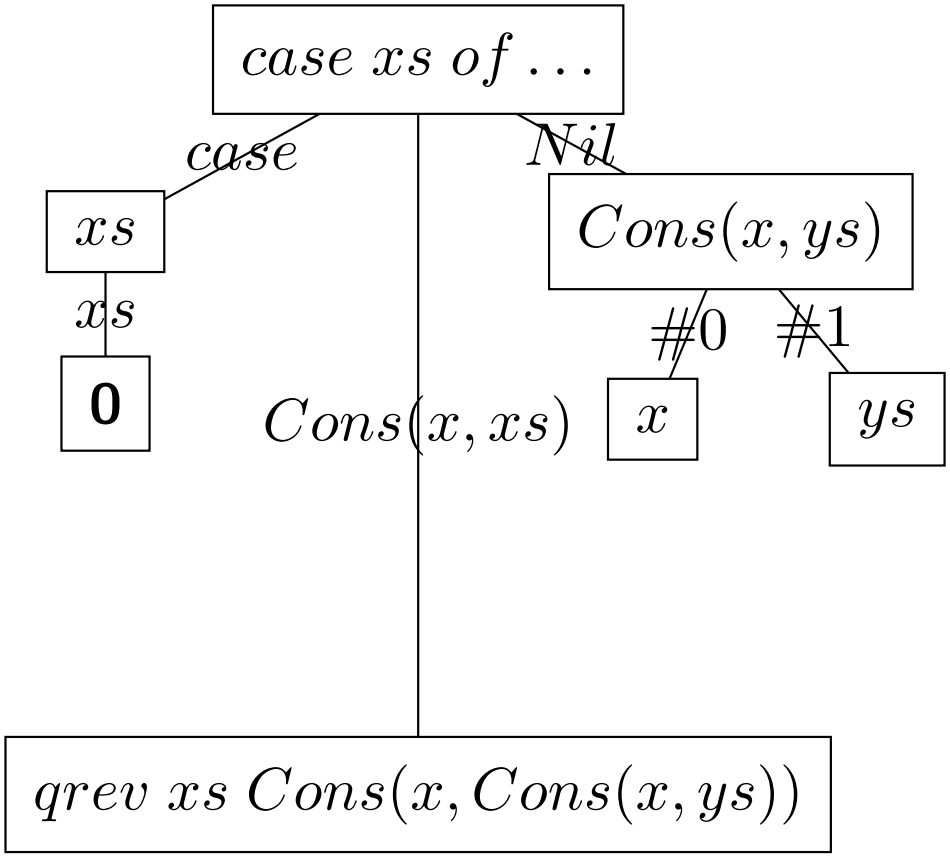
Техника дистилляции: построение помеченной системы переходов по программе

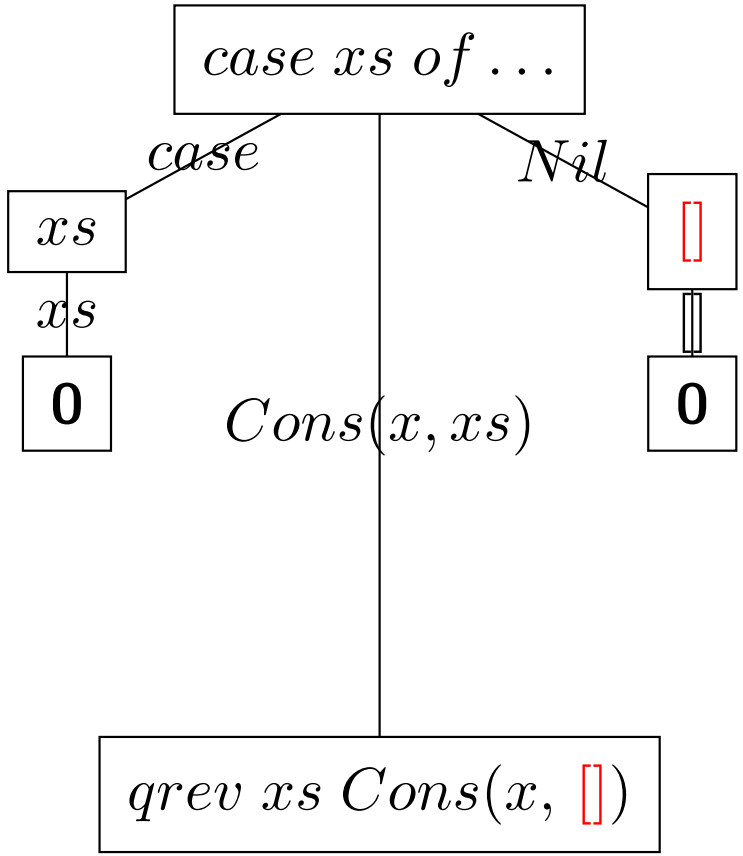


Техника дистилляции: построение помеченной системы переходов по программе



Техника дистилляции: обобщение





Техника дистилляции: обобщение