24.04.20 ФМ-5 Лекция 4

Трансформации системы Why3

http://why3.lri.fr/manual.pdf
http://why3.lri.fr/stdlib/

Стандартная библиотека

Трансформации.

Организация

Приведение к нормальному виду.

Подстановка определений.

Декомпозиция гипотез

Расщепление

Применение теорем

Действия с кванторами

Стандартная библиотека Why3

algebra : Basic Algebra Theories array : Arrays bag : Multisets (aka bags)
bintree : Polymorphic binary trees with elements at nodes
bool : Booleans bv : Bit Vectors exn : General-purpose exceptions
floating_point : Formalization of Floating-Point Arithmetic
fmap : Finite Maps function : Injections, surjections and bijections
graph : Graph theory hashtbl : Hash tables
int : Theory of integers list : Polymorphic Lists map : Theory of maps
matrix : Matrices null : A possibly null, yet safe, value number : Number theory
option : Option type ocaml : General functions related to OCaml extraction
pigeon : Pigeon hole principle pqueue : Priority queues
queue : Polymorphic mutable queues random : Pseudo-random generators
real : Theory of reals ref : References regexp : Theory of regular expressions
relations : Relations seq : Sequences set : Set theories stack : Stacks
string : Theory of strings tree : Polymorphic n-ary trees
mach.array : Arrays with bounded-size integers
mach.bv : Program functions on bitvectors with preconditionsenforcing absence of overflow .....

Стандартная библиотека Why3

Импорт теорий из стандартной библиотеки

```
use int.Int
use array.Array
```

Импорт теорий машинной арифметики

```
use mach.int.Int63
use mach.array.Array63
```

Организация трансформаций Why3

Трансформация – команда, применяемая к текущей доказываемой формуле (цели). В результате получается другая формула или несколько других целей.

Имя трансформации. Параметры трансформации.

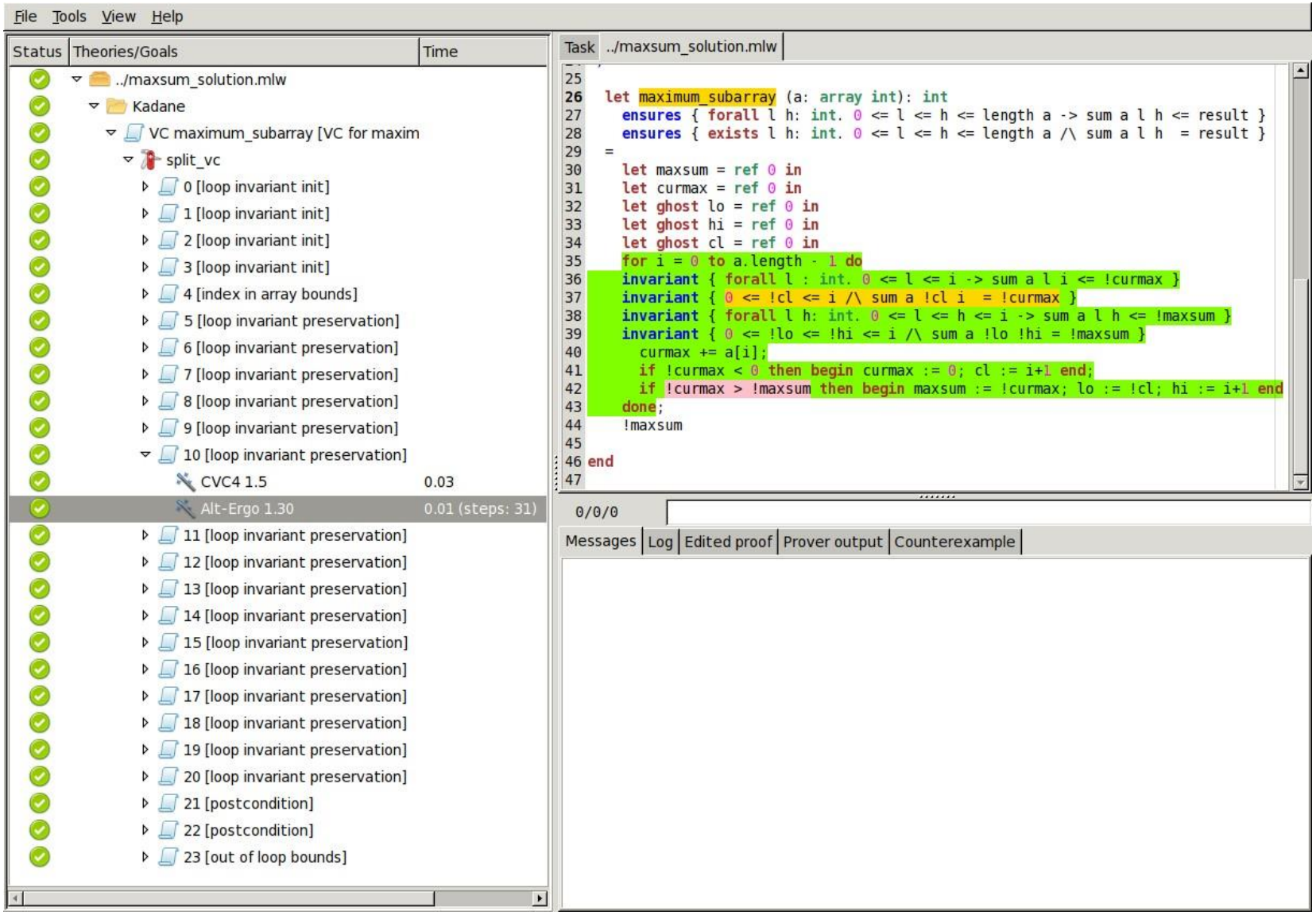
Перечень доступных трансформаций:

why3 --list-transforms

Запуск трансформаций из командной строки либо через раздел Tool в верхнем меню

Трансформации упорядочены по алфавиту

Why3 proof session



Приведение цели к нормальному виду

Нажать раздел **task** в верхнем правом меню.

Внизу – доказываемая формула (цель), выше - контекст

Контекст: определения типов, констант, аксиом, функций, предикатов, … в проекции на цель.

Гипотезы (аксиомы): посылки + подкванторные переменные – появляются непосредственно выше после нормализации цели. Имя гипотезы

introduce_premises – полная нормализация цели

intros <имена переменных через запятую > – вынесение подкванторных переменных с назначением имен соответствующих констант

intros_n <целое> – вынесение указанного числа подкванторных переменных

Подстановка определения

Подстановка тела нерекурсивного определения на место вызова функции (предиката) с заменой вхождений формальных параметров на соответствующие фактические параметры

inline_goal – для всех вызовов только внутри цели

inline_all – везде, в цели и гипотезах

inline_trivial – подставить и удалить определения вида

function f x_1 ... x_n = g e_1 ... e_k

predicate p x_1 ... x_n = q e_1 ... e_k

e_i – простой или x_j, каждое x_1 ... x_n встречается не более 1 раза

unfold <имя определения> [**in** <имя гипотезы>] – подстановка определения с данным именем в цели или указанной гипотезе

Декомпозиция гипотез

destruct <имя гипотезы> – устраняет конструкцию (операцию) верхнего уровня в указанной гипотезе.

Применяется к конструкциям вида:

- **false**, **true**,
- /\, \/, ->, **not**,
- **exists**,
- **if** ... **then** ... **else** ...,
- **match** ... **with** ... **end**,
- (не)равенство выражений одного типа.

destruct_rec <имя гипотезы> – многократное применение трансформации **destruct**

destruct_term <имя переменной> – декомпозиция по альтернативам (конструктурам) типа переменной

eliminate_if_term – вынос **if** . **then** . **else** на верхний уровень

split_premise_right – гипотезы в конъюнктивной форме заменяются несколькими гипотезами

split_premise_full – предварительно приводятся в конъюнктивную форму

Расщепление

split_goal_right – цель в виде конъюнкции формул заменяется набором целей для каждого конъюнкта исходной цели.

Цель вида A && B заменяется целями A и A -> B

Цель вида A \/ B -> C заменяется целями A -> C и B -> C

Цель вида A || B -> C заменяется на A -> C и (**not** A) /\ B -> C

Цели вида **if** , **match** ...

split_goal_full – предварительно преобразуется в конъюнктивную форму

split_all_right , **split_all_full** – дополнительно применяется **split_premise**

case (<формула>) – расщепляет на две цели для истинного и ложного значений <формулы>

case (<формула>) **as** <имя гипотезы>

Расщепление при **destruct** и других трансформациях

Применение теорем

apply <имя гипотезы> – применение правила modus ponens. Пусть: h: f -> p; G: p. Трансформация **apply** h заменяет исходную цель на G: f . Для подкванторных переменных в h определяется замена на подходящие термы.

apply <имя гипотезы> **with** <терм> – трансформация с подсказкой.

assert (<формула>) – вводит новую цель, с помощью которой доказывается исходная цель. Для цели G: p трансформация **assert** (f) дает две цели h: f и G: f -> p

assert (<формула>) **as** <имя гипотезы>

cut (<формула>) – как и **assert,** но с другим порядком генерируемых целей

Действия с кванторами

exists <терм> – подстановка в цели <терма> на место переменной под квантором существования

instantiate <имя гипотезы> <список термов через запятую> – создается новая гипотеза с заменой кванторных переменных указанными термами.

inst_rem <имя гипотезы> <список термов через запятую> – как **instantiate** с удалением исходной гипотезы

Дизъюнктивная форма. Замены

left – удаление правой части дизъюнктивной формы

right – удаление левой части дизъюнктивной формы

pose <имя> <терм> – введение новой гипотезы вида <имя> = <терм>

remove <список имен гипотез> – удалить указанные гипотезы

replace <терм1> <терм2> [**in** <имя гипотезы>] – в цели или гипотезе <терм1> заменяется на <терм2>. Генерируется дополнительная цель: <терм1> = <терм2>

rewrite <имя гипотезы> - заключительная часть гипотезы имеет вид <терм1> = <терм2>. В цели <терм1> меняется на <терм2>. Посылки гипотезы дают новые цели. Для подстановки подкванторных переменных используется подсказка вида **with** <терм>

Доказательство по индукции

induction <имя переменной> – доказательство по индукции для текущей цели: h : p1 n, G: p n. Команда **induction** n дает две цели.

h : p1 n, Init: n<=0, G: p n

h : p1 n, Init: n>0, hRect: **forall** n1:int. n1 < n -> p1 n1 -> p n1, G: p n

Предварительно следует свернуть зависимость от n

revert – собрать гипотезы в цель, обратная к intros

induction_ty_lex <имя> – для алгебраических типов

Особенности доказательства

Преимущества последней версии. Новые трансформации

Оператор assert { … }

Лемма-функции вместо доказательства по индукции

Возможности полного доказательства в Why3.

Перевод доказательства в Coq

Список трансформаций отдельным файлом