Verification in Isabelle/HOL of Hopcroft's algorithm for minimizing DFAs including runtime analysis

Vincent Trélat
supervised by Prof. Dr. Tobias Nipkow and Dr. Peter Lammich

Technical University of Munich
Chair for Logic and Verification

September 2023

Outline

Figure: Location of Munich

Figure: Some photos of Munich

Figure: Technical University of Munich (TUM), Garching campus

Figure: Technical University of Munich (TUM), Garching campus


Kloster
Scheyern
KLOSTER·GOLD
DUNKEL


Kloster Scheyern
SEIT 1119
KLOSTER·GOLD
DUNKEL


Edelstoff
13 28
J. W.
Augustiner-
Bräu München
Edelstoff


Kloster
Scheyern
SEIT 1119
KLOSTER-GOLD
DUNKEL


Edelstoff
13 28
J. W.
Augustiner-
Bräu München
Edelstoff


Kloster
Scheyern
KLOSTER·GOLD
DUNKEL


Edelstoff
J.W.
Augustiner-
Bräu München
Edelstoff


Kloster
Scheyern
KLOSTER·GOLD
DUNKEL


Kloster
Scheyern
KLOSTER-GOLD
DUNKEL


Edelstoff
J.W.
Augustiner-
Bräu München
Edelstoff


Kloster
Scheyern
KLOSTER-GOLD
DUNKEL


SPEZIAL
HERZOGLICH BAYERISCHES
BRAUHAUS
TEGERNSEE


KLOSTER-GOLD
DUNKEL
Kloster
Scheyern
KLOSTER-GOLD
DUNKEL


Edelstoff
13 28
J.W.
Augustiner
Bräu München
Edelstoff


KLOSTER-GOLD
DUNKEL
Kloster
Scheyern
KLOSTER-GOLD
DUNKEL


SPEZIAL
BRAUHAUS
TEGERNSEE


Kloster
Scheyern
KLOSTER-GOLD
DUNKEL


Edelstoff
J.W.
Augustiner-
Bräu München
Edelstoff


Kloster
Scheyern
KLOSTER-GOLD
DUNKEL


SPEZIAL
BRAUHAUS
TEGERNSEE


WIRTS MADL
Bayrisch Hell


Kloster Scheyern
KLOSTER-GOLD
DUNKEL


Edelstoff
J.W.
Augustiner Bräu München
Edelstoff


Kloster Scheyern
KLOSTER-GOLD
DUNKEL


SPEZIAL
BRAUHAUS
TEGERNSEE


WIRTS MADL
Bayrisch Hell


HB

Understanding Refinement

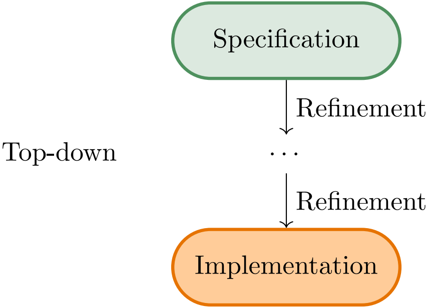
Definition 1

Refinement is a systematic process of refining a high-level abstract specification into a concrete implementation.

Understanding Refinement

Definition 1

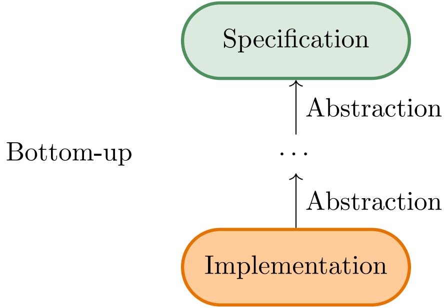
Refinement is a systematic process of refining a high-level abstract specification into a concrete implementation.



Understanding Refinement

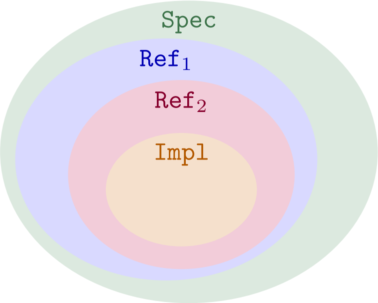
Definition 1

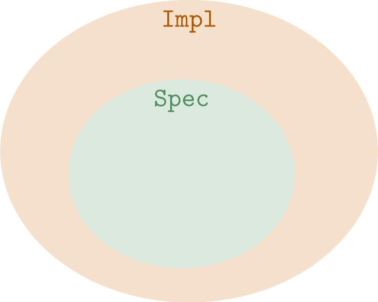
Refinement is a systematic process of refining a high-level abstract specification into a concrete implementation.



Each refinement step preserves the intended behavior.

$$\texttt{Spec} \rightarrow \texttt{Ref}_1 \rightarrow \texttt{Ref}_2 \rightarrow \texttt{Impl}$$



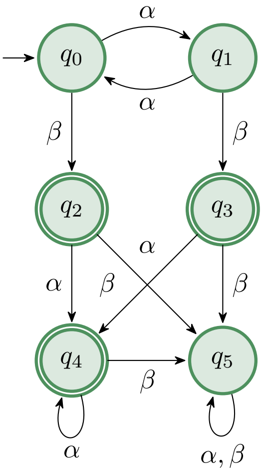


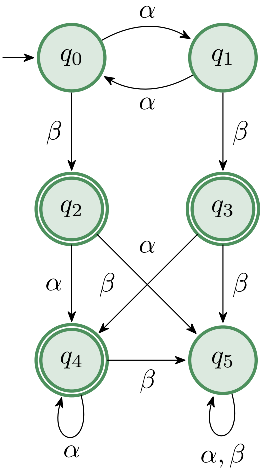
The Isabelle Refinement Framework

Isabelle Refinement Framework

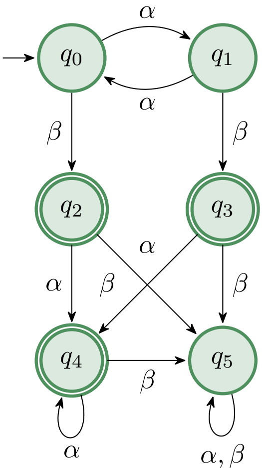
- Stepwise refinement approach to verified program development
- Formal and mathematical
- Ensures correctness at each step

Comes with the Isabelle Collection Framework, which provides an extensive library of reusable verified functional data structures (for data refinement).

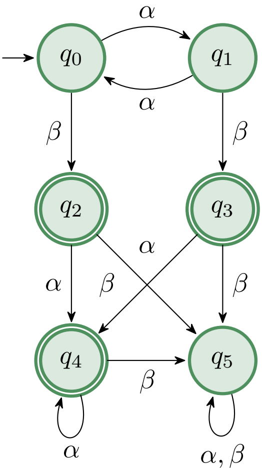

α
q_0
q_1
α
β
β
q_2
q_3
α
α
β
β
q_4
q_5
β
α
α, β



- Successively partitions the set of states into equivalence classes



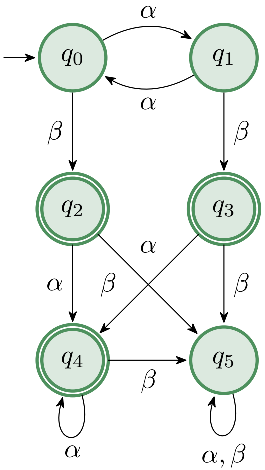
- Successively partitions the set of states into equivalence classes
- Initial partition: accepting and non-accepting states



- Successively partitions the set of states into equivalence classes
- Initial partition: accepting and non-accepting states
- Each iteration: pick a splitter and split all blocks of the current partition

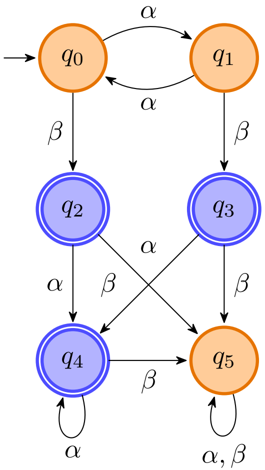
Data: a DFA $\mathcal{A} = (\mathcal{Q}, \Sigma, \delta, q_0, \mathcal{F})$
if $\mathcal{F} = \varnothing \vee \mathcal{Q} \setminus \mathcal{F} = \varnothing$
  return $\mathcal{Q}$
else

$\mathcal{P} := \{\mathcal{F}, \mathcal{Q} \setminus \mathcal{F}\}$
$\mathcal{W} := \{(a, \min\{\mathcal{F}, \mathcal{Q} \setminus \mathcal{F}\}), a \in \Sigma\}$
while $\mathcal{W} \neq \varnothing$ **do**
  Pick (a, C) from $\mathcal{W}$ and remove it
  forall $B \in \mathcal{P}$ **do**
    Split B with (a, C) into B_0 and B_1
    $\mathcal{P} := (\mathcal{P} \setminus \{B\}) \cup \{B_0, B_1\}$
    forall $b \in \Sigma$ **do**
      if $(b, B) \in \mathcal{W}$
        $\mathcal{W} :=$
         $(\mathcal{W} \setminus \{(b, B)\}) \cup \{(b, B_0), (b, B_1)\}$
      else
        $\mathcal{W} := \mathcal{W} \cup \{(b, \min\{B_0, B_1\})\}$



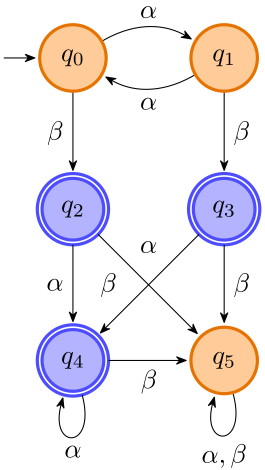
Data: a DFA $\mathcal{A} = (\mathcal{Q}, \Sigma, \delta, q_0, \mathcal{F})$
if $\mathcal{F} = \varnothing \vee \mathcal{Q} \setminus \mathcal{F} = \varnothing$
  return $\mathcal{Q}$
else

$\mathcal{P} := \{\mathcal{F}, \mathcal{Q} \setminus \mathcal{F}\}$
$\mathcal{W} := \{(a, \min\{\mathcal{F}, \mathcal{Q} \setminus \mathcal{F}\}), a \in \Sigma\}$
while $\mathcal{W} \neq \varnothing$ **do**
  Pick (a, C) from $\mathcal{W}$ and remove it
  forall $B \in \mathcal{P}$ **do**
    Split B with (a, C) into B_0 and B_1
    $\mathcal{P} := (\mathcal{P} \setminus \{B\}) \cup \{B_0, B_1\}$
    forall $b \in \Sigma$ **do**
      if $(b, B) \in \mathcal{W}$
        $\mathcal{W} :=$
        $(\mathcal{W} \setminus \{(b, B)\}) \cup \{(b, B_0), (b, B_1)\}$
      else
        $\mathcal{W} := \mathcal{W} \cup \{(b, \min\{B_0, B_1\})\}$



Splitter	Partition	Workset
–	$\{q_0, q_1, q_5\}\{q_2, q_3, q_4\}$	$(\alpha, \{q_0, q_1, q_5\})$ $(\beta, \{q_0, q_1, q_5\})$

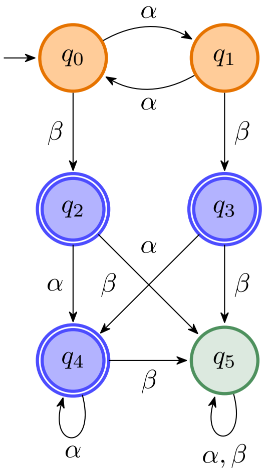
Data: a DFA $\mathcal{A} = (\mathcal{Q}, \Sigma, \delta, q_0, \mathcal{F})$
if $\mathcal{F} = \varnothing \vee \mathcal{Q} \setminus \mathcal{F} = \varnothing$
  return $\mathcal{Q}$
else
$\mathcal{P} := \{\mathcal{F}, \mathcal{Q} \setminus \mathcal{F}\}$
$\mathcal{W} := \{(a, \min\{\mathcal{F}, \mathcal{Q} \setminus \mathcal{F}\}), a \in \Sigma\}$
while $\mathcal{W} \neq \varnothing$ **do**
  Pick (a, C) from $\mathcal{W}$ and remove it
  forall $B \in \mathcal{P}$ **do**
    Split B with (a, C) into B_0 and B_1
    $\mathcal{P} := (\mathcal{P} \setminus \{B\}) \cup \{B_0, B_1\}$
    forall $b \in \Sigma$ **do**
      if $(b, B) \in \mathcal{W}$
        $\mathcal{W} :=$
        $(\mathcal{W} \setminus \{(b, B)\}) \cup \{(b, B_0), (b, B_1)\}$
      else
        $\mathcal{W} := \mathcal{W} \cup \{(b, \min\{B_0, B_1\})\}$



Splitter	Partition	Workset
–	$\{q_0, q_1, q_5\}\{q_2, q_3, q_4\}$	$(\alpha, \{q_0, q_1, q_5\})$ $(\beta, \{q_0, q_1, q_5\})$

Data: a DFA $\mathcal{A} = (\mathcal{Q}, \Sigma, \delta, q_0, \mathcal{F})$
if $\mathcal{F} = \varnothing \vee \mathcal{Q} \setminus \mathcal{F} = \varnothing$
  return $\mathcal{Q}$
else

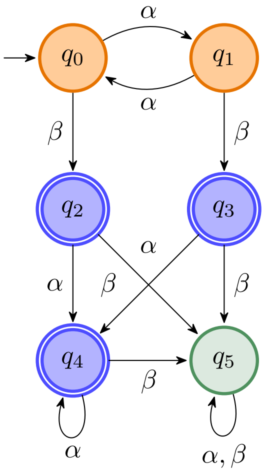
$\mathcal{P} := \{\mathcal{F}, \mathcal{Q} \setminus \mathcal{F}\}$
$\mathcal{W} := \{(a, \min\{\mathcal{F}, \mathcal{Q} \setminus \mathcal{F}\}), a \in \Sigma\}$
while $\mathcal{W} \neq \varnothing$ **do**
  Pick (a, C) from $\mathcal{W}$ and remove it
  forall $B \in \mathcal{P}$ **do**
    Split B with (a, C) into B_0 and B_1
    $\mathcal{P} := (\mathcal{P} \setminus \{B\}) \cup \{B_0, B_1\}$
    forall $b \in \Sigma$ **do**
      if $(b, B) \in \mathcal{W}$
        $\mathcal{W} :=$
         $(\mathcal{W} \setminus \{(b, B)\}) \cup \{(b, B_0), (b, B_1)\}$
      else
        $\mathcal{W} := \mathcal{W} \cup \{(b, \min\{B_0, B_1\})\}$



Splitter	Partition	Workset
–	$\{q_0, q_1, q_5\}\{q_2, q_3, q_4\}$	$(\alpha, \{q_0, q_1, q_5\})$ $(\beta, \{q_0, q_1, q_5\})$
$(\beta, \{q_0, q_1, q_5\})$	$\{q_0, q_1\}\{q_5\}\{q_2, q_3, q_4\}$	$(\alpha, \{q_0, q_1\})$ $(\alpha, \{q_5\})$

Data: a DFA $\mathcal{A} = (\mathcal{Q}, \Sigma, \delta, q_0, \mathcal{F})$
if $\mathcal{F} = \varnothing \vee \mathcal{Q} \setminus \mathcal{F} = \varnothing$
  return $\mathcal{Q}$
else

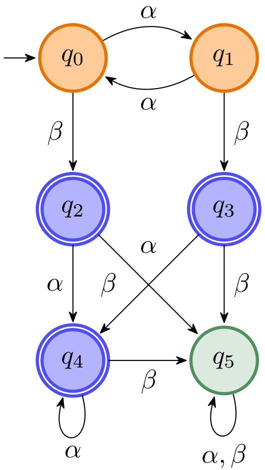
$\mathcal{P} := \{\mathcal{F}, \mathcal{Q} \setminus \mathcal{F}\}$
$\mathcal{W} := \{(a, \min\{\mathcal{F}, \mathcal{Q} \setminus \mathcal{F}\}), a \in \Sigma\}$
while $\mathcal{W} \neq \varnothing$ **do**
  Pick (a, C) from $\mathcal{W}$ and remove it
  forall $B \in \mathcal{P}$ **do**
    Split B with (a, C) into B_0 and B_1
    $\mathcal{P} := (\mathcal{P} \setminus \{B\}) \cup \{B_0, B_1\}$
    forall $b \in \Sigma$ **do**
      if $(b, B) \in \mathcal{W}$
        $\mathcal{W} :=$
         $(\mathcal{W} \setminus \{(b, B)\}) \cup \{(b, B_0), (b, B_1)\}$
      else
        $\mathcal{W} := \mathcal{W} \cup \{(b, \min\{B_0, B_1\})\}$



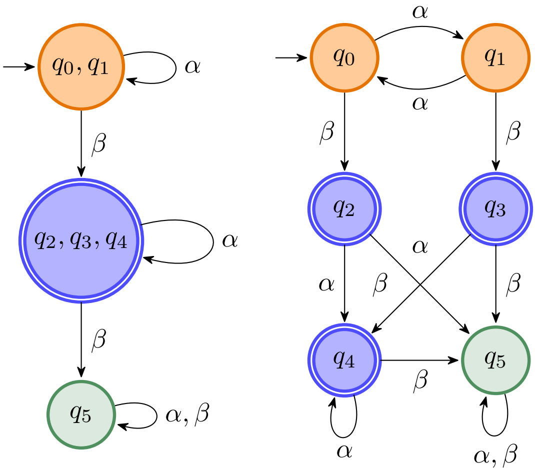
Splitter	Partition	Workset
–	$\{q_0, q_1, q_5\}\{q_2, q_3, q_4\}$	$(\alpha, \{q_0, q_1, q_5\})$ $(\beta, \{q_0, q_1, q_5\})$
$(\beta, \{q_0, q_1, q_5\})$	$\{q_0, q_1\}\{q_5\}\{q_2, q_3, q_4\}$	$(\alpha, \{q_0, q_1\})$ $(\alpha, \{q_5\})$
$(\alpha, \{q_0, q_1\})$	$\{q_0, q_1\}\{q_5\}\{q_2, q_3, q_4\}$	$(\alpha, \{q_5\})$

Data: a DFA $\mathcal{A} = (\mathcal{Q}, \Sigma, \delta, q_0, \mathcal{F})$
if $\mathcal{F} = \varnothing \vee \mathcal{Q} \setminus \mathcal{F} = \varnothing$
  return $\mathcal{Q}$
else

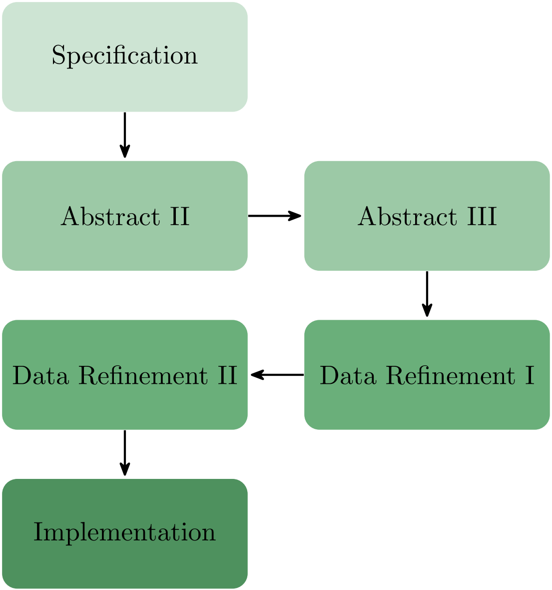
$\mathcal{P} := \{\mathcal{F}, \mathcal{Q} \setminus \mathcal{F}\}$
$\mathcal{W} := \{(a, \min\{\mathcal{F}, \mathcal{Q} \setminus \mathcal{F}\}), a \in \Sigma\}$
while $\mathcal{W} \neq \varnothing$ **do**
  Pick (a, C) from $\mathcal{W}$ and remove it
  forall $B \in \mathcal{P}$ **do**
    Split B with (a, C) into B_0 and B_1
    $\mathcal{P} := (\mathcal{P} \setminus \{B\}) \cup \{B_0, B_1\}$
    forall $b \in \Sigma$ **do**
      if $(b, B) \in \mathcal{W}$
        $\mathcal{W} :=$
         $(\mathcal{W} \setminus \{(b, B)\}) \cup \{(b, B_0), (b, B_1)\}$
      else
        $\mathcal{W} := \mathcal{W} \cup \{(b, \min\{B_0, B_1\})\}$

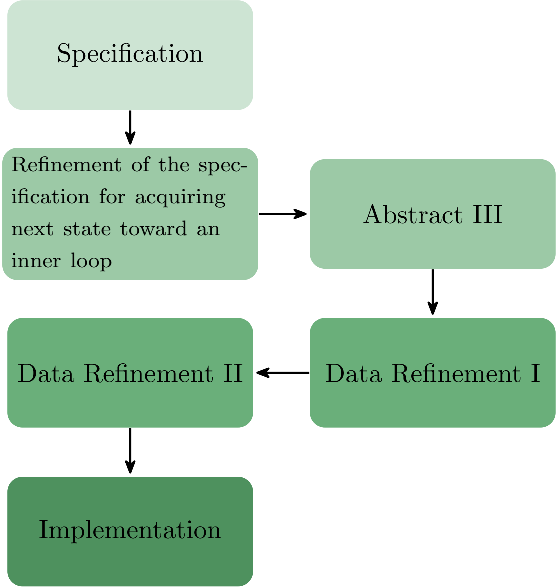


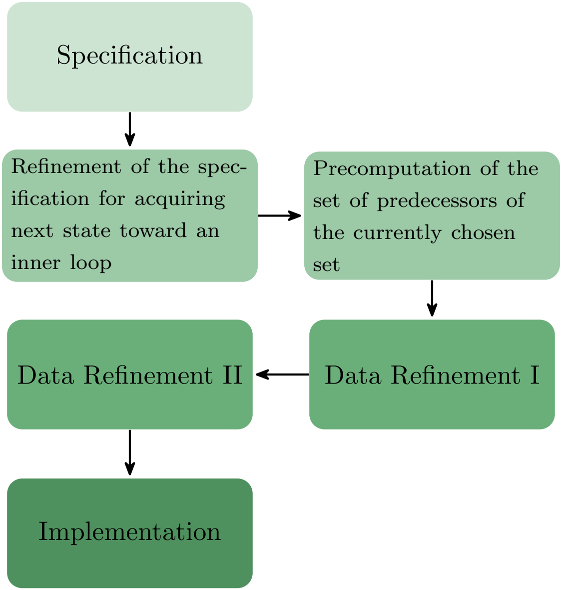
Splitter	Partition	Workset
–	$\{q_0, q_1, q_5\}\{q_2, q_3, q_4\}$	$(\alpha, \{q_0, q_1, q_5\})$ $(\beta, \{q_0, q_1, q_5\})$
$(\beta, \{q_0, q_1, q_5\})$	$\{q_0, q_1\}\{q_5\}\{q_2, q_3, q_4\}$	$(\alpha, \{q_0, q_1\})$ $(\alpha, \{q_5\})$
$(\alpha, \{q_0, q_1\})$	$\{q_0, q_1\}\{q_5\}\{q_2, q_3, q_4\}$	$(\alpha, \{q_5\})$
$(\alpha, \{q_5\})$	$\{q_0, q_1\}\{q_5\}\{q_2, q_3, q_4\}$	$\varnothing$

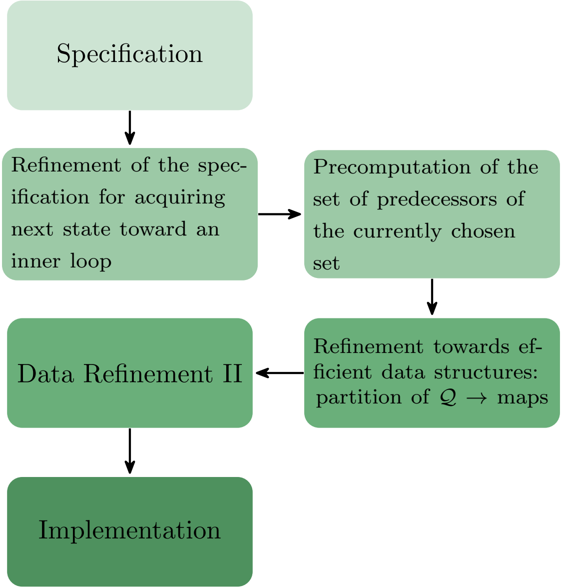


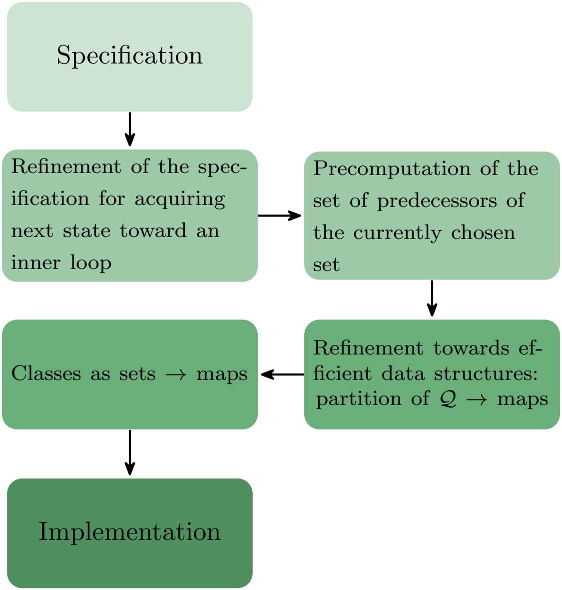
$$\mathcal{L} = \alpha^* \beta \alpha^*$$

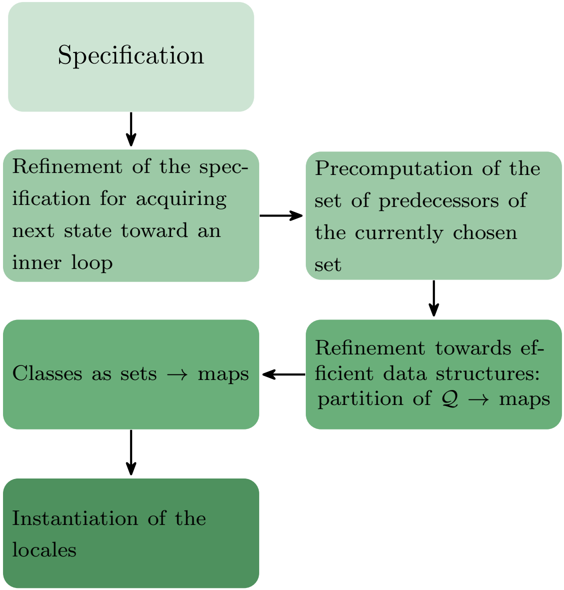

Specification
Abstract II
Abstract III
Data Refinement I
Data Refinement II
Implementation


Specification
Refinement of the specification for acquiring next state toward an inner loop
Abstract III
Data Refinement I
Data Refinement II
Implementation


Specification
Refinement of the specification for acquiring next state toward an inner loop
Precomputation of the set of predecessors of the currently chosen set
Data Refinement I
Data Refinement II
Implementation


Specification
Refinement of the specification for acquiring next state toward an inner loop
Precomputation of the set of predecessors of the currently chosen set
Refinement towards efficient data structures: partition of $\mathcal{Q}$ → maps
Data Refinement II
Implementation


Specification
Refinement of the specification for acquiring next state toward an inner loop
Precomputation of the set of predecessors of the currently chosen set
Refinement towards efficient data structures: partition of $\mathcal{Q} \rightarrow$ maps
Classes as sets $\rightarrow$ maps
Implementation


Specification
Refinement of the specification for acquiring next state toward an inner loop
Precomputation of the set of predecessors of the currently chosen set
Refinement towards efficient data structures: partition of $\mathcal{Q} \to$ maps
Classes as sets $\to$ maps
Instantiation of the locales

Specification

Refinement of the specification for acquiring next state toward an inner loop

Precomputation of the set of predecessors of the currently chosen set

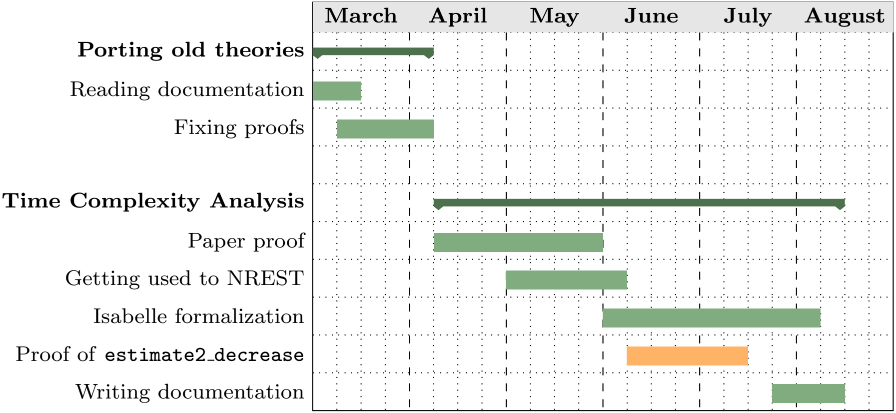
What I had time to do

Refinement towards efficient data structures: partition of $\mathcal{Q} \to$ maps

Classes as sets $\to$ maps

Instantiation of the locales

Gantt chart


March
April
May
June
July
August
Porting old theories
Reading documentation
Fixing proofs
Time Complexity Analysis
Paper proof
Getting used to NREST
Isabelle formalization
Proof of estimate2_decrease
Writing documentation

Statistics about the project: