情報システム学特別講義1
「高階モデル検査とプログラム検証」

東京大学
小林　直樹

講義の計画

- 第一部
  - 高階モデル検査とその意義
- 第二部
  - 高階モデル検査アルゴリズム
- 第三部
  - 高階モデル検査のプログラム検証への応用

第二部の目次

- ♦高階モデル検査問題
  - 高階再帰スキーム (済)
  - 木オートマトン
    - 自明オートマトン
    - 交代性パリティ木オートマトン
- ♦高階モデル検査の型判定問題への帰着
- ♦高階モデル検査アルゴリズム

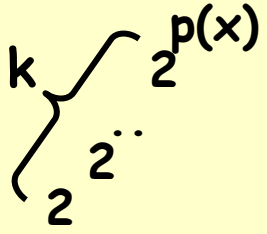
高階モデル検査問題

入力:
- G: 高階再帰スキーム(HORS)
- A: (無限木を受理する)木オートマトン
  (交代性パリティ木オートマトン、
  または様相μ計算の論理式),

出力: A が Gの生成する木Tree(G)を受理するか否か

検証できる性質の例
- Does every finite path end with "c"?
- Does "a" occur below "b"?

定理 (Ong 2006)
order-k HORSに対する
高階モデル検査問題はk重指数完全



（有限語を受理する）オートマトン(復習）

$A = (\Sigma, Q, \Delta, q_0, F)$

Σ: 入力記号の有限集合　　Q: 状態の有限集合

q_0: 初期状態　　Δ: 遷移規則($q \to_a q'$)の集合

F: 受理状態の集合

A が$w=a_1 \ldots a_n$を受理

$\Leftrightarrow \exists q_1 \ldots q_n. (q_0 \to_{a1} q_1 \to_{a2} \ldots \to_{an} q_n \in F)$

例：偶数個のaを持つ語を受理するオートマトン

$(\{a,b\}, \{even,odd\}, \Delta, even, \{even\})$

$\Delta = \{even \to_a odd,\ odd \to_a even,$
$even \to_b even,\ odd \to_b odd\}$

$even \to_a odd \to_b odd \to_b odd \to_a even$なのでabbaは受理

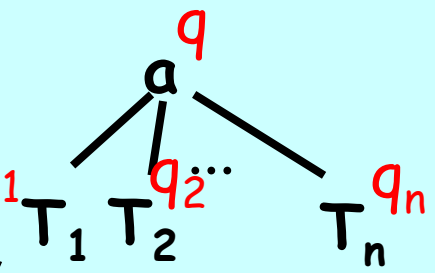
(トップダウン)非決定性木オートマトン

$A = (\Sigma, Q, \Delta, q_0)$

Σ: 入力記号の有限集合(arityつき)

Q: 状態の有限集合　　q_0: 初期状態

Δ: 遷移規則($q \rightarrow_a q_1 \ \ldots \ q_{arity(a)}$)の集合



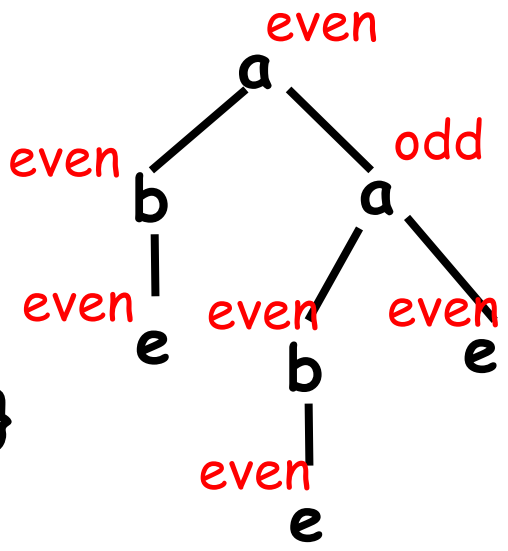
A が木Tを受理

⇔ Tの各ノードに遷移規則に従った状態割り当てが可能

例:aを偶数個含む木を受理するオートマトン

({a/2,b/1,e/0}, {even,odd}, Δ, even)

Δ = {even $\rightarrow_a$odd even, even $\rightarrow_a$even odd,
odd $\rightarrow_a$even even, odd $\rightarrow_a$odd odd,
even $\rightarrow_b$even, odd $\rightarrow_b$odd, even $\rightarrow_e$ }



(トップダウン)交代性木オートマトン

$A = (\Sigma, Q, \delta, q_0)$

Σ: 入力記号の有限集合(arityつき)

Q: 状態の有限集合　　q_0: 初期状態

δ: 遷移関数

(状態、入力記号の組を(i,q_i), $\wedge$, $\vee$, true, falseからなる論理式に写像)

A が木Tを受理

$\Leftrightarrow$ 遷移規則に従ったオートマトンの状態遷移、分岐が可能

例:aを偶数個含む木を受理する交代性木オートマトン

$(\{a/2, b/1, e/0\}, \{even, odd\}, \delta, even)$

$\delta(even, a) = ((1,even) \wedge (2,odd)) \vee ((1,odd) \wedge (2,even))$

$\delta(odd, a) = ((1,even) \wedge (2,even)) \vee ((1,odd) \wedge (2,odd))$

$\delta(even, b) = (1,even)$　　$\delta(odd, b) = (1, odd)$

$\delta(even, e) = true$　　$\delta(odd, e) = true$

無限語を受理する非決定性木オートマトン

非決定性木オートマトン＋受理条件

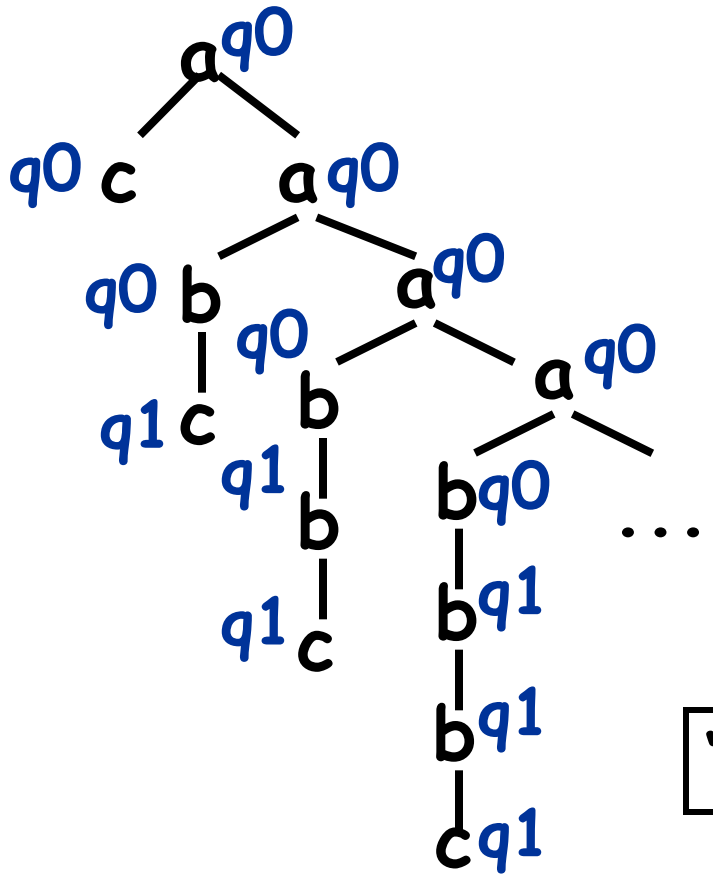
A が木Tを受理

⇔ 遷移規則に従うTの各ノードに対する状態割り当てのうち、すべての無限パス上の状態列が受理条件を満たすものあり

オートマトン(*)	受理条件
自明木オートマトン	なし（自明）
Büchi 木オートマトン	受理状態の少なくとも一つが無限回出現
パリティ木オートマトン	無限回現れる状態の優先度の最大値が偶数

(*) 他に代表的なものとして Muller , Streett, Rabinオートマトンがある

自明木オートマトンの例



$q0 \rightarrow_a q0\ q0$

$q0 \rightarrow_b q1$

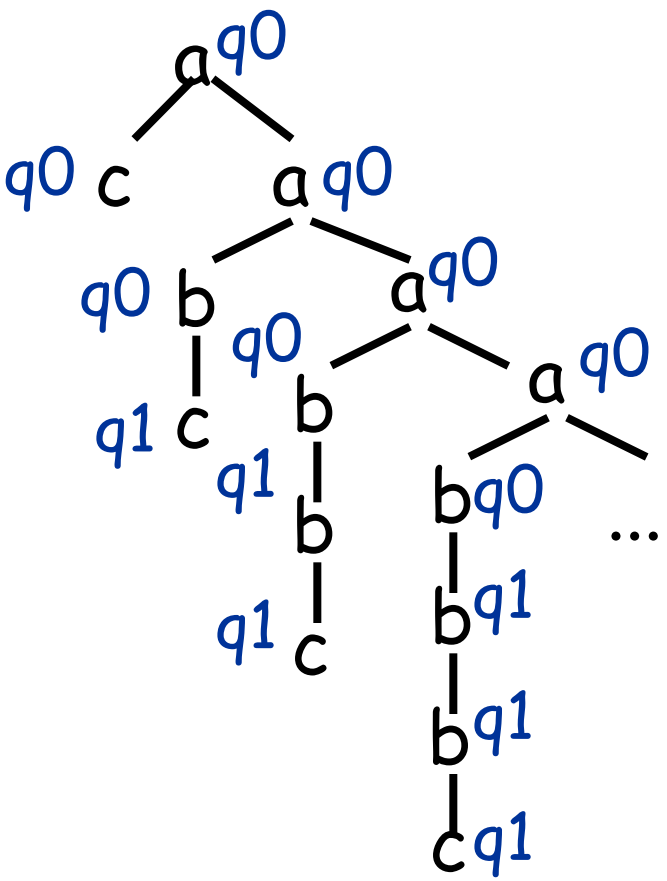
$q1 \rightarrow_b q1$

$q0 \rightarrow_c \varepsilon$

$q1 \rightarrow_c \varepsilon$

"a" does not occur below "b"

(非決定性)パリティ木オートマトンの例



$q0 \rightarrow_a q0\ q0$
$q0 \rightarrow_b q1$
$q1 \rightarrow_b q1$
$q0 \rightarrow_c \varepsilon$
$q1 \rightarrow_c \varepsilon$

優先度関数:
$\Omega(q0) = 1$
$\Omega(q1) = 2$

受理条件: すべてのパスにおいて、無限回出現する状態の優先度の最大値が偶数　= すべての無限パスはbを含む

無限木に対するオートマトンの表現力

- 遷移関数による分類

  決定性 < 非決定性 = 交代性

  （ただし交代性の方が同じ性質を小さく表現可能）

- 受理条件による分類

  自明 < Büchi < パリティ (=Muller=Streett= Rabin)

- safety property の表現には自明木オートマトンで十分

- 交代性パリティ木オートマトンと様相μ計算の論理式との間では線形サイズの相互変換が可能 [Kupferman+,2000]

練習問題

- アルファベット{a/2,b/1,c/1}から構成される無限木で、次の条件を満たすもののみを受理するパリティ木オートマトンを作れ。

  「すべてのパスにおいて、
  (bが無限回出現するならば
  cも無限回出現する)」

練習問題解答例

$(\{a/2, b/1, c/1\}, \{q_a, q_b, q_c\}, D, q_a, \Omega)$

$q \rightarrow_a q_a\ q_a$　　$q \rightarrow_b q_b$　　$q \rightarrow_c q_c$

(for every q)

$\Omega(q_a)=0,\ \Omega(q_b)=1,\ \Omega(q_c)=2$

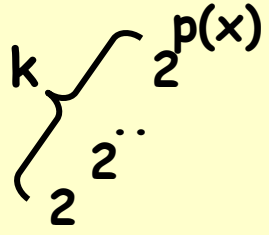
第二部の目次

- ♦高階モデル検査問題
  - 高階再帰スキーム
  - 木オートマトン
- ♦高階モデル検査の型判定問題への帰着
  - 自明木オートマトンの場合
  - パリティ木オートマトンの場合
- ♦高階モデル検査アルゴリズム

高階モデル検査問題

入力:
　G:　高階再帰スキーム(HORS)
　A:　(無限木を受理する)木オートマトン
　　(交代性パリティ木オートマトン、
　　または様相μ計算の論理式),
出力: A が Gの生成する木Tree(G)を受理するか否か

定理 (Ong 2006)
order-k HORSに対する
高階モデル検査問題は**k重指数完全**



目標

♦ 高階モデル検査と等価な型システムを構成

Tree(G) is accepted by A

if and only if

$\vdash_A G$　(GがAによってパラメタ化された型システム$\vdash_A$で型づけ可能)

➔高階モデル検査の決定可能性の平易な証明

➔効率的な高階モデル検査アルゴリズム

既存の高階モデル検査アルゴリズム

- 第0世代(決定可能性の証明のための理論的なもの。常にn重指数時間がかかる)
  - ゲーム意味論に基づくもの[Ong, LICS06]
  - CPDS経由 [Hague+,LICS08]
  - 型理論に基づくもの[K, POPL09][K&Ong, POPL09]
- 第1世代:
  - TRecS アルゴリズム[K, PPDP09]
  - GTRecS [K, FoSSaCS11]
  - TravMC [Neatherway+, 12]
- 第2世代:
  - HorSat [Broadbent&K, CSL13]
  - PREFACE [Ramsay+, POPL14]
  - CSHORe [Hague+, ICFP13]

型判定問題への帰着[K, POPL09]を利用

第二部の目次

- 高階モデル検査問題
  - 高階再帰スキーム
  - 木オートマトン
- 高階モデル検査の型判定問題への帰着
  - 自明木オートマトンの場合
  - パリティ木オートマトンの場合
- 高階モデル検査アルゴリズム

高階モデル検査問題(一般の場合)

入力:
　G:　高階再帰スキーム(HORS)
　A:　(交代性)パリティ木オートマトン
出力: A が Gの生成する木Tree(G)を受理するか否か

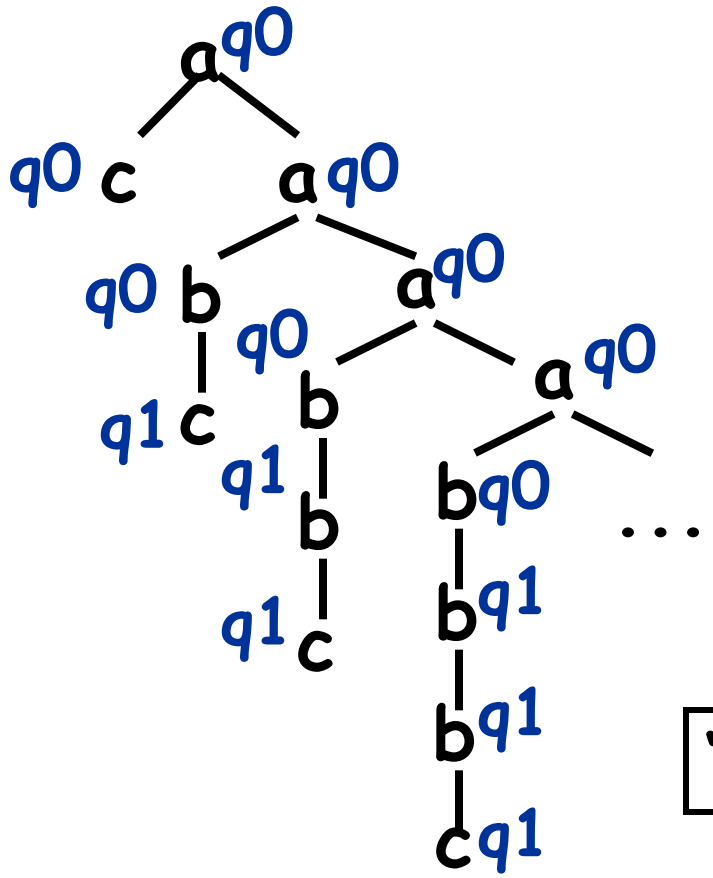
定理 (Ong 2006)
order-k HORSに対する
高階モデル検査問題はk重指数完全

制限つき高階モデル検査問題

入力:
　G:　高階再帰スキーム(HORS)
　A:　自明木オートマトン
出力: A が Gの生成する木Tree(G)を受理するか否か

定理 (K&Ong 2009)
- order-k HORSに対する制限つき
  高階モデル検査問題はk重指数完全
- 自明木オートマトンをさらに決定性に制限すると
  (k-1)重指数完全

自明木オートマトンの例（再掲）

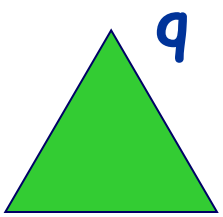


$q0 \rightarrow_a q0\ q0$
$q0 \rightarrow_b q1$
$q1 \rightarrow_b q1$
$q0 \rightarrow_c \varepsilon$
$q1 \rightarrow_c \varepsilon$

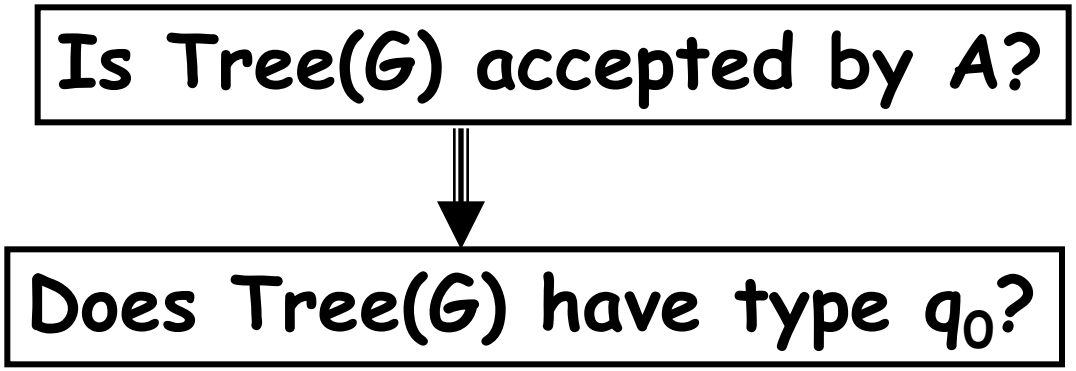
"a" does not occur below "b"

型システムのアイデア

- オートマトンの状態を木の型とみなす
  - q：状態qから受理される木の型



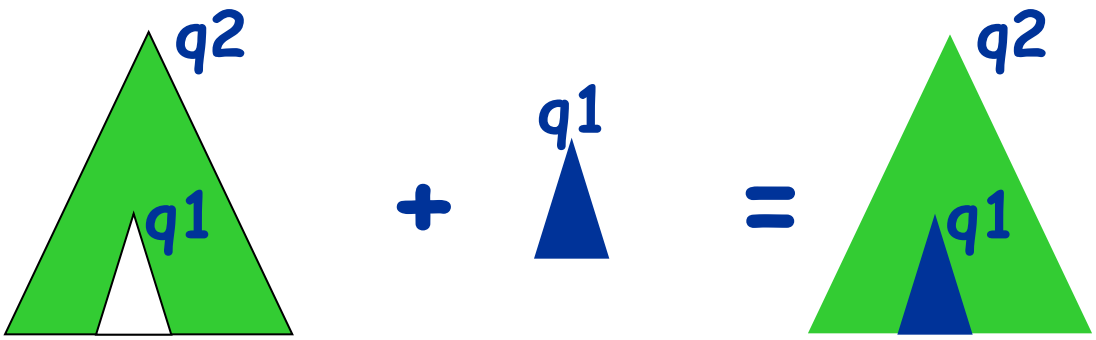
  - q1∧q2：状態q1とq2の両方から受理できる木の型



木に関する関数(文脈)の型

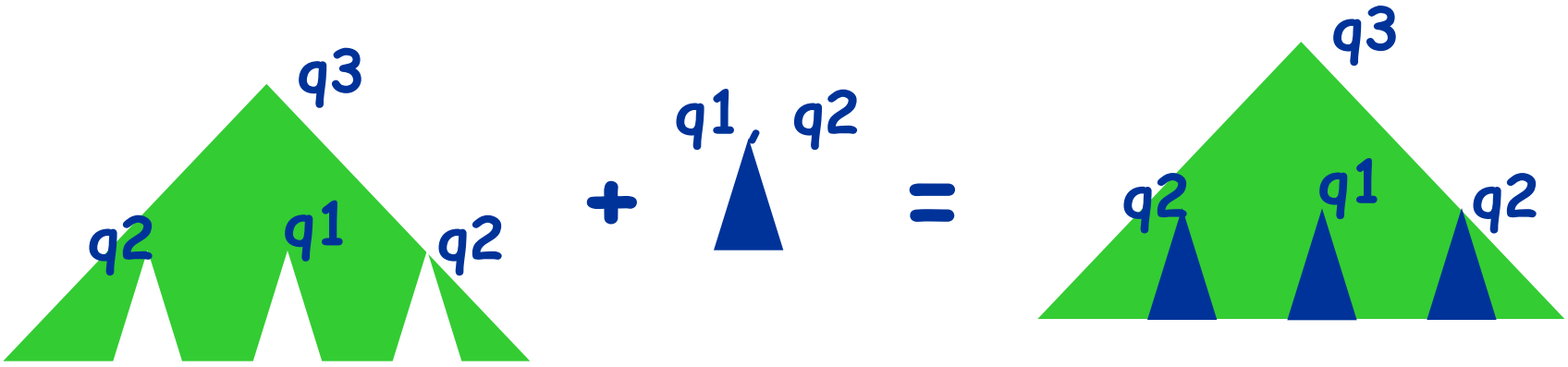
q1→ q2:

functions that take a tree of type q1 and return a tree of q2



木に関する関数(文脈)の型

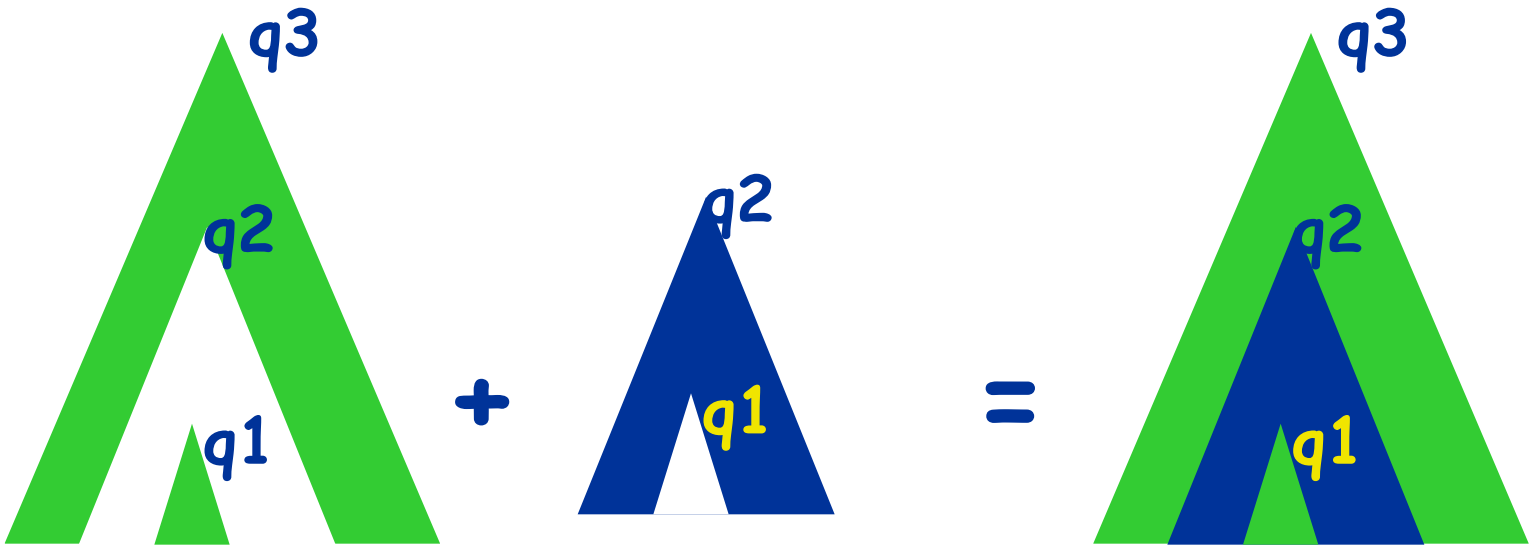
$q1 \wedge q2 \rightarrow q3$:
functions that take a tree of type $q1 \wedge q2$
and return a tree of type $q3$



木に関する高階関数(文脈)の型

(q1 → q2) → q3:

functions that take a function of type q1 → q2
and return a tree of type q3



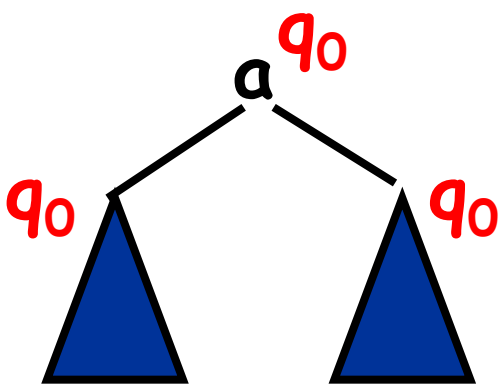
Example

Automaton:

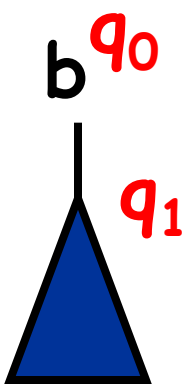
$q_0 \to_a q_0\ q_0$ $\quad q_0 \to_b q_1$ $\quad q_1 \to_b q_1$

$q_0 \to_c \varepsilon$ $\quad q_1 \to_c \varepsilon$

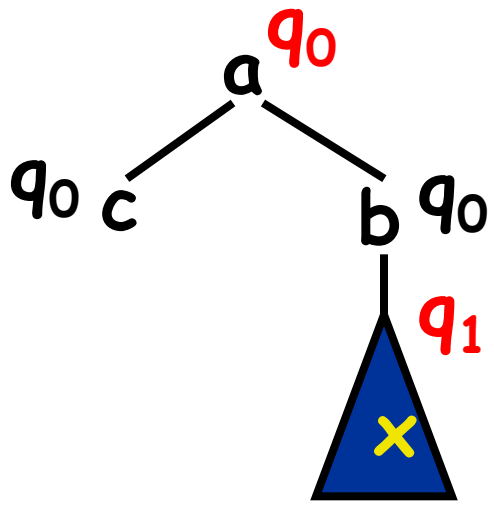
a: $q_0 \to q_0 \to q_0$



b: $q_1 \to q_0$



$\lambda x.a\ c\ (b\ x)$: $q_1 \to q_0$



型付け規則

$$\frac{q \to_a q_1 \ldots q_n \in \Delta}{\vdash a : q_1 \to \ldots \to q_n \to q}$$

$$\Gamma, x{:}\tau \vdash x : \tau$$

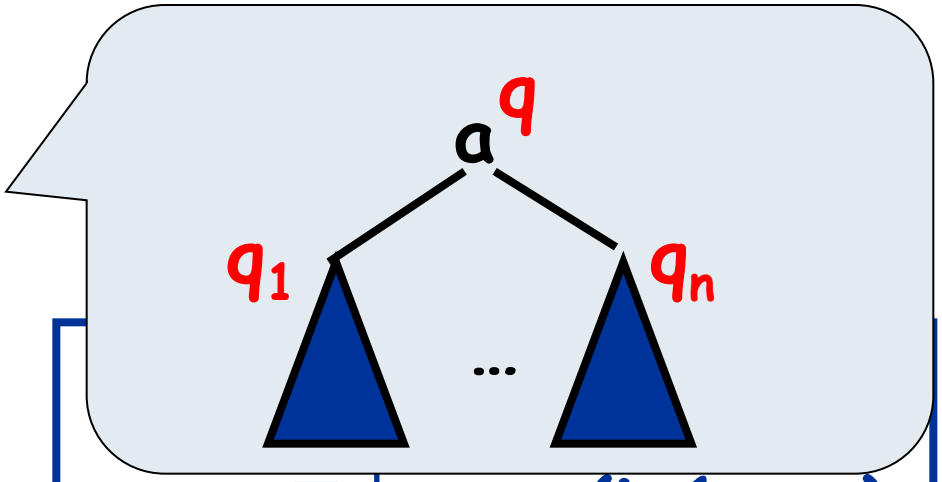
$$\frac{\Gamma, x{:}\tau_1, \ldots, x{:}\tau_n \vdash t{:}\tau}{\Gamma \vdash \lambda x.t{:}\ \tau_1 \wedge \ldots \wedge \tau_n \to \tau}$$

$$\frac{\Gamma \vdash t_1{:}\ \tau_1 \wedge \ldots \wedge \tau_n \to \tau \qquad \Gamma \vdash t_2{:}\tau_i\ (i=1,..n)}{\Gamma \vdash t_1\ t_2{:}\tau}$$

$$\frac{\Gamma \vdash t_k : \tau\ (\text{for every } F_k{:}\tau \in \Gamma)}{\vdash \{F_1 \to t_1, \ldots, F_n \to t_n\} : \Gamma}$$

型付け規則

$$\frac{q \to_a q_1 \ldots q_n \in \Delta}{\vdash a : q_1 \to \ldots \to q_n \to q}$$



$$\frac{\Gamma, x:\tau_1, \ldots, x:\tau_n \vdash t:\tau}{\Gamma \vdash \lambda x.t: \tau_1 \wedge \ldots \wedge \tau_n \to \tau}$$

$$\frac{\Gamma \vdash t_2:\tau_i \ (i=1,..n)}{\Gamma \vdash t_1\ t_2:\tau}$$

$$\frac{\Gamma \vdash t_k : \tau \ (\text{for every } F_k:\tau \in \Gamma)}{\vdash \{F_1 \to t_1, \ldots, F_n \to t_n\} : \Gamma}$$

型の導出例

Automaton:

$q_0 \to_a q_0\ q_0$ $\quad q_0 \to_b q_1$ $\quad q_1 \to_b q_1$

$q_0 \to_c \varepsilon$ $\quad q_1 \to_c \varepsilon$

$$\dfrac{\dfrac{\dfrac{\vdash a: q_0 \to q_0 \to q_0 \quad x:q_0 \vdash x: q_0}{x:q_0 \vdash a\ x: q_0 \to q_0} \quad \dfrac{\vdash b: q_1 \to q_0 \quad x:q_1 \vdash x: q_1}{x:q_1 \vdash (b\ x): q_0}}{x:q_0,\ x:q_1 \vdash a\ x\ (b\ x): q_0}}{\vdash \lambda x.a\ x\ (b\ x): q_0 \wedge q_1 \to q_0}$$

型付け規則

$$\frac{q \to_a q_1 \ldots q_n \in \Delta}{\vdash a : q_1 \to \ldots \to q_n \to q}$$

$$\Gamma, x{:}\tau \vdash x : \tau$$

$$\frac{\Gamma, x{:}\tau_1, \ldots, x{:}\tau_n \vdash t{:}\tau}{\Gamma \vdash \lambda x.t{:}\ \tau_1 \wedge \ldots \wedge \tau_n \to \tau}$$

$$\frac{\Gamma \vdash t_1{:}\ \tau_1 \wedge \ldots \wedge \tau_n \to \tau \qquad \Gamma \vdash t_2{:}\tau_i\ (i=1,..n)}{\Gamma \vdash t_1\ t_2{:}\tau}$$

$$\frac{\Gamma \vdash t_k : \tau\ (\text{for every } F_k{:}\tau \in \Gamma)}{\vdash \{F_1 \to t_1, \ldots, F_n \to t_n\} : \Gamma}$$

書き換え規則の型導出例

HORS: $S \to F\ c \qquad F \to \lambda x.a\ x\ (F\ (b\ x))$

Automaton:

$q_0 \to_a q_0\ q_0 \qquad q_0 \to_b q_1 \qquad q_1 \to_b q_1$

$q_0 \to_c \varepsilon \qquad q_1 \to_c \varepsilon$

$S{:}q_0,\ F{:}\ q_0 \wedge q_1 \to q_0 \vdash \lambda x.a\ x\ (F\ (b\ x)){:}\ q_0 \wedge q_1 \to q_0$

$S{:}q_0,\ F{:}\ q_0 \wedge q_1 \to q_0 \vdash F\ c : q_0$

$\vdash \{S \to F\ c,\ F \to \lambda x.a\ x\ (F\ (b\ x))\}{:}\ \{S{:}q_0,\ F{:}\ q_0 \wedge q_1 \to q_0\}$

型システムの健全性と完全性

[K., POPL2009]

Tree(G) is accepted by $A^{\perp}$
if and only if
S has type q_0 in TS(A),
i.e. $\exists \Gamma.(S{:}q_0 \in \Gamma \wedge \vdash \{F_1 \to t_1, \ldots, F_n \to t_n\} : \Gamma)$

$G = \{F_1 \to t_1, \ldots, F_m \to t_m\}$ (with $S=F_1$)
A: Trivial automaton with initial state q_0
TS(A): Intersection type system for A
$A^{\perp}$: Extension of A with rule $q \to_{\perp} \varepsilon$ for each q

第二部の目次

- 高階モデル検査問題
- 高階モデル検査の型判定問題への帰着
- 高階モデル検査アルゴリズム
  - ナイーブな不動点計算アルゴリズム
  - 実践的アルゴリズム(TRecSアルゴリズム)
  - 最新の動向

型システムの健全性と完全性 (自明オートマトンの場合)

Tree(G) is accepted by A
if and only if

S has type q_0 in TS(A),

i.e. $\exists\Gamma.(S:q_0 \in \Gamma \wedge \vdash \{F_1 \to t_1, \ldots, F_n \to t_n\} : \Gamma)$

$G = \{F_1 \to t_1, \ldots, F_m \to t_m\}$ (with $S=F_1$)
A: Trivial automaton with initial state q_0
TS(A): Intersection type system for A

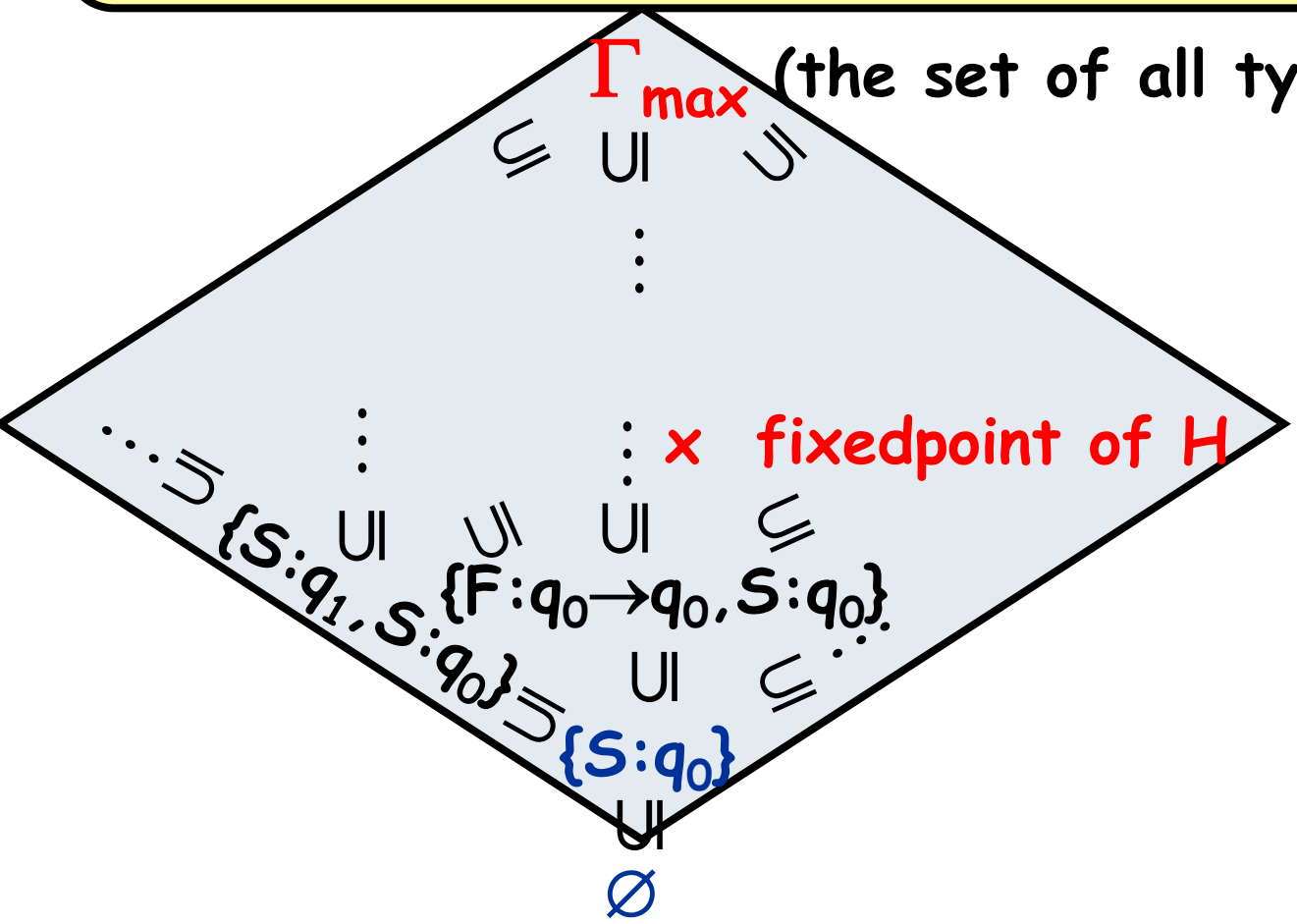
型システムの健全性と完全性

Tree(G) is accepted by A
if and only if
S has type q_0 in TS(A),
i.e. $\exists\Gamma.(S{:}q_0 \in \Gamma \wedge \vdash \{F_1 \to t_1, \ldots, F_n \to t_n\} : \Gamma)$
if and only if
$\exists\Gamma.(S{:}\ q_0 \in \Gamma \wedge \forall(F_k{:}\tau)\in\Gamma.\ \Gamma \vdash t_k : \tau\ )$
if and only if
$\exists\Gamma.(S{:}\ q_0 \in \Gamma \wedge \Gamma = H(\Gamma)\ )$
for $H(\Gamma) = \{\ F_k{:}\tau \in \Gamma \mid \Gamma \vdash t_k{:}\tau\ \}$

Function to filter out invalid type bindings

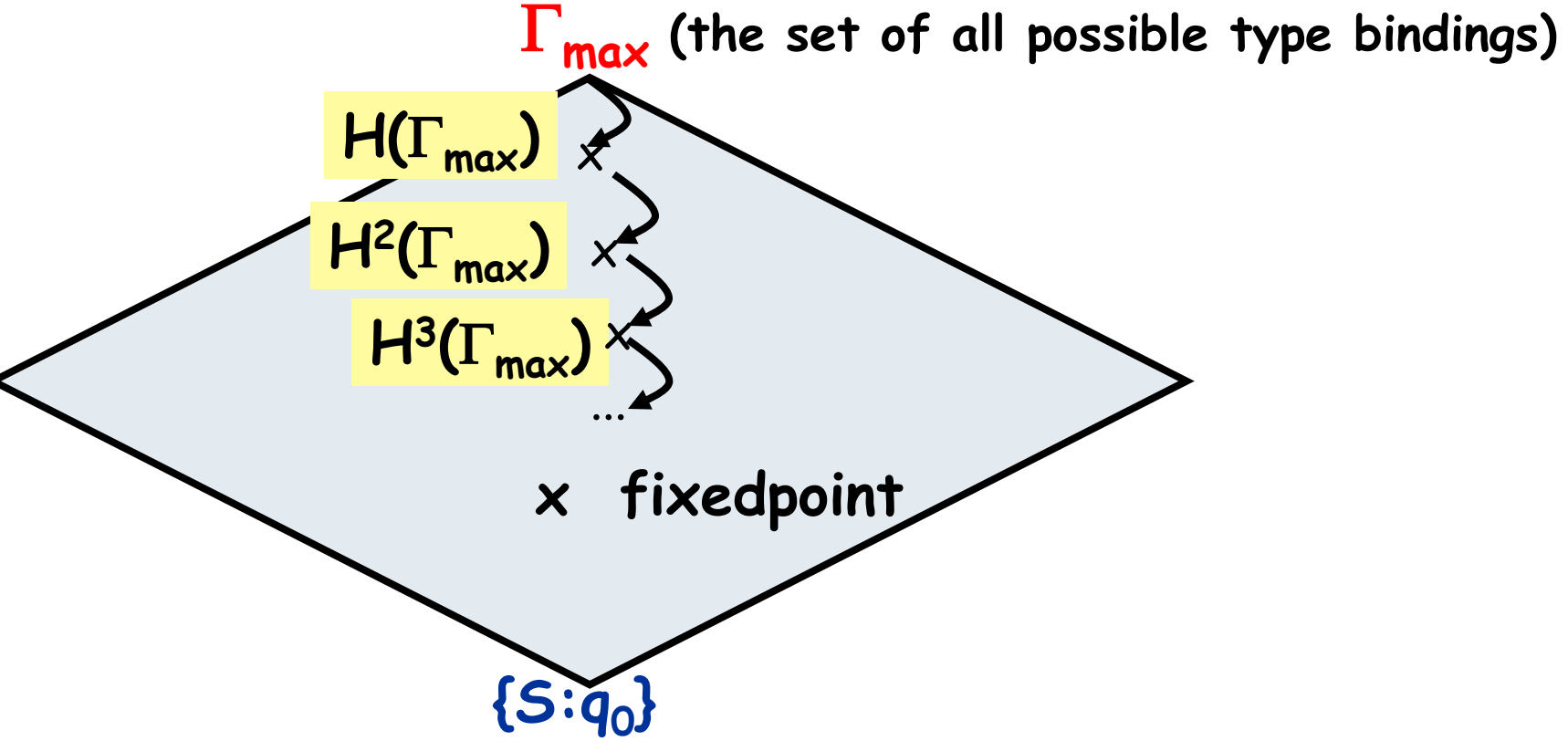
モデル検査問題=型検査問題=不動点問題

Is there a fixedpoint of H greater than $\{S:q_0\}$?
(where $H(\Gamma) = \{ F_j:\tau \in \Gamma \mid \Gamma \mid\!- t_j:\tau \}$)



Naive Algorithm [K. POPL09]

1. Compute the greatest fixedpoint Γ_{gfp} of H
   ($H(\Gamma) = \{ F_j:\tau \in \Gamma \mid \Gamma \mid\!- t_j:\tau \}$)
2. Check whether $S:q_0 \in \Gamma_{gfp}$



Example

- HORS: $S \to F\ c \qquad F \to \lambda x.a\ x\ (F\ (b\ x))$

  $(S{:}o,\ F{:}\ o \to o)$

- Automaton: $q_0 \to_a q_0\ q_0 \qquad q_0 \to_b q_1 \qquad q_1 \to_b q_1$

  $q_0 \to_c \varepsilon \qquad q_1 \to_c \varepsilon$

$\Gamma_{max} = \{S{:}q_0,\ S{:}q_1,\ F{:}T \to q_0,\ F{:}\ T \to q_1,\ F{:}q_0 \to q_0,\ F{:}q_1 \to q_0,$
$F{:}\ q_0 \wedge q_1 \to q_0,\ F{:}\ q_0 \to q_1,\ F{:}\ q_1 \to q_1,\ F{:}\ q_0 \wedge q_1 \to q_1\}$

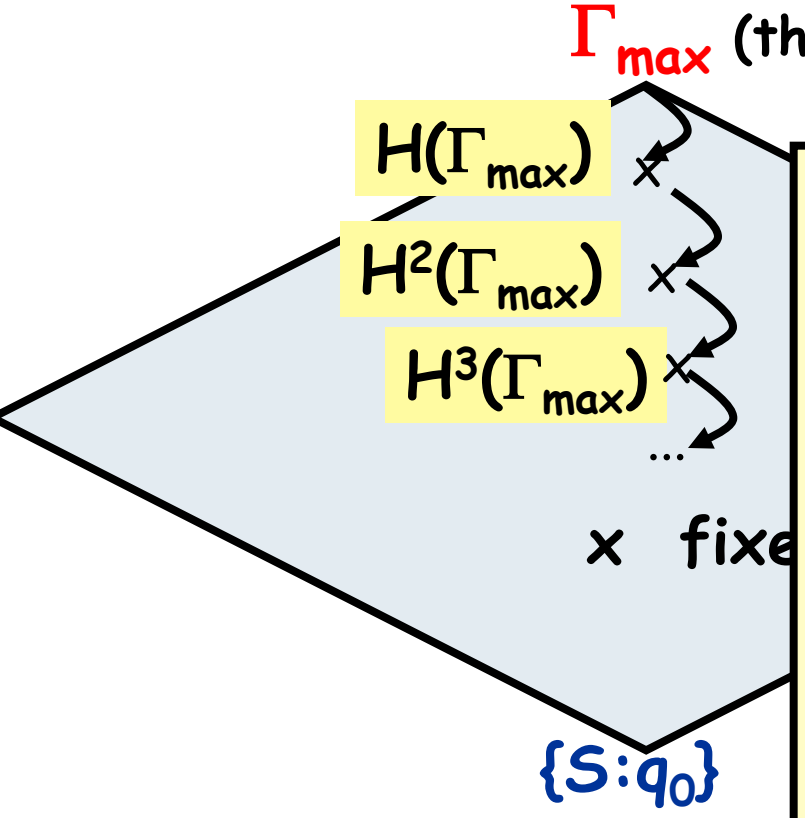
$H(\Gamma_{max}) = \{\ S{:}\tau \in \Gamma_{max} \mid \Gamma_{max} \vdash F\ c{:}\tau\ \}$
$\cup\ \{\ F{:}\tau \in \Gamma_{max} \mid \Gamma_{max} \vdash \lambda x.a\ x\ (F(b\ x))\ {:}\tau\ \}$
$= \{S{:}q_0,\ S{:}q_1,\ \ F{:}\ q_0 \to q_0,\ F{:}\ q_0 \wedge q_1 \to q_0\}$

$H^2(\Gamma_{max}) = \{S{:}q_0,\ F{:}\ q_0 \wedge q_1 \to q_0\}$

$H^3(\Gamma_{max}) = \{S{:}q_0,\ F{:}\ q_0 \wedge q_1 \to q_0\} = H^2(\Gamma_{max})$

Naive Algorithm [K. POPL09]

1. Compute the greatest fixedpoint Γ_{gfp} of H
   ($H(\Gamma) = \{ F_j:\tau \in \Gamma \mid \Gamma \mid\!- t_j:\tau \}$)
2. Check whether $S:q_0 \in \Gamma_{gfp}$



Drawbacks:
- Huge cost for computing H
- Huge number of iterations

(both as huge as $|\Gamma_{max}| = (AQ)^{1+\varepsilon}$

$$O(|G| \times k^{2^{2^{\cdot^{\cdot^{2}}}}})$$

A: largest arity
Q: automaton size

How large is Γ_{max}?

Γ_{max}: the set of all possible type bindings for non-terminals

sort	# of types for each sort ($Q=\{q_0,q_1,q_2,q_3\}$)
o (trees)	4 (q_0,q_1,q_2,q_3)
$o \to o$	$2^4 \times 4 = 64$ $(\wedge S \to q$, with $S \in 2^Q$, $q \in Q)$
$(o \to o) \to o$	$2^{64} \times 4 = 2^{66}$
$((o \to o) \to o) \to o$	$2^{2^{66}} \times 4 > 10^{10000000000000000000}$

$$|\Gamma_{max}| = O(|G| \times \underbrace{2^{2^{\cdot^{\cdot^{2}}}}}_{k}{}^{(A|Q|)^{1+\varepsilon}})$$

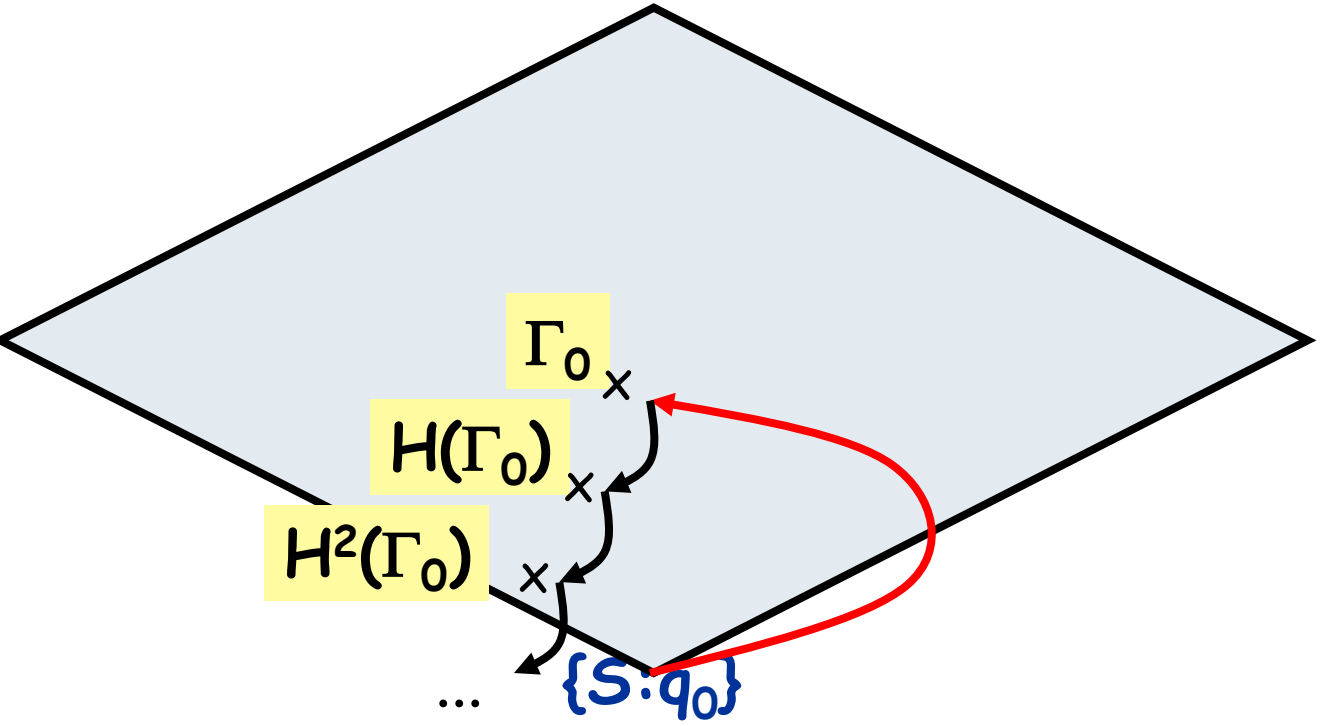
第二部の目次

- 高階モデル検査問題
- 高階モデル検査の型判定問題への帰着
- 高階モデル検査アルゴリズム
  - ナイーブな不動点計算アルゴリズム
  - 実践的アルゴリズム(TRecSアルゴリズム)
  - 最新の動向

Practical Algorithms [K. PPDP09] [K.FoSSaCS11]

1. Guess a type environment Γ_0
2. Compute greatest fixedpoint Γ smaller than Γ_0
3. Check whether $S{:}q_0 \in \Gamma$
4. Repeat 1-3 until the property is proved or refuted.

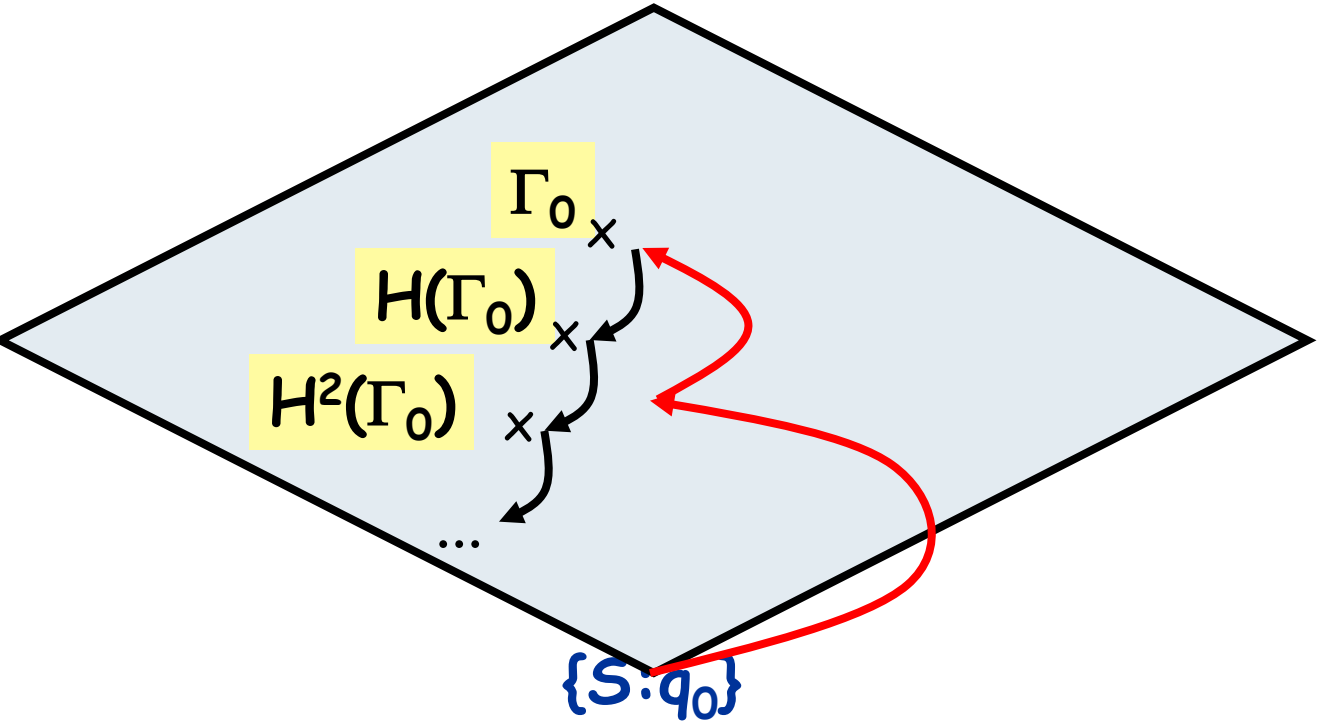
Γ_{max} (the set of all possible type bindings)



Practical Algorithms [K. PPDP09] [K.FoSSaCS11]

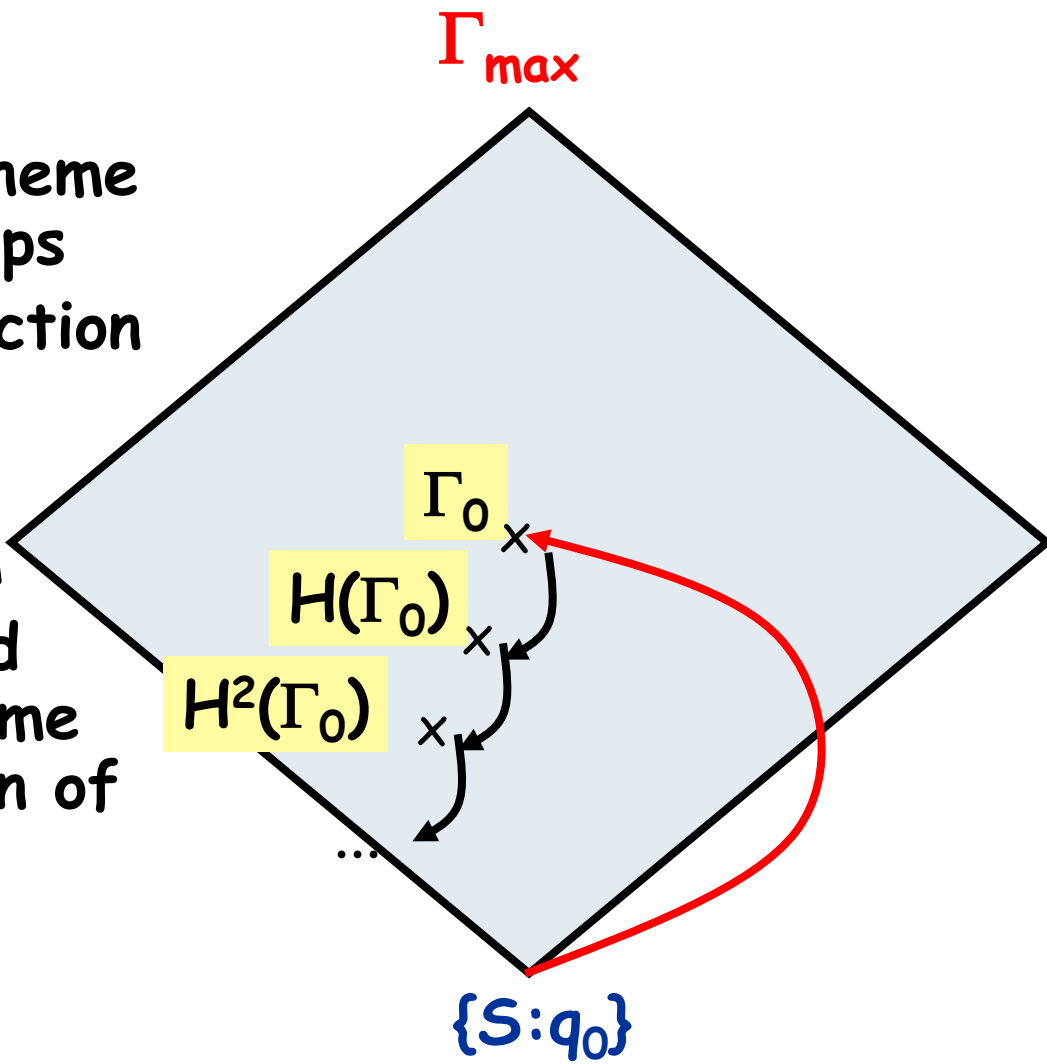
1. Guess a type environment Γ_0
2. Compute greatest fixedpoint Γ smaller than Γ_0
3. Check whether $S{:}q_0 \in \Gamma$
4. Repeat 1-3 until the property is proved or refuted.

Γ_{max} (the set of all possible type bindings)



How to guess Γ_0?

- PPDP09 algorithm
  - Reduce a recursion scheme a finite number of steps
  - Observe how each function is used and express it as types
- FoSSaCS11 algorithm
  - Like PPDP09, but avoid reductions by using game semantic interpretation of types



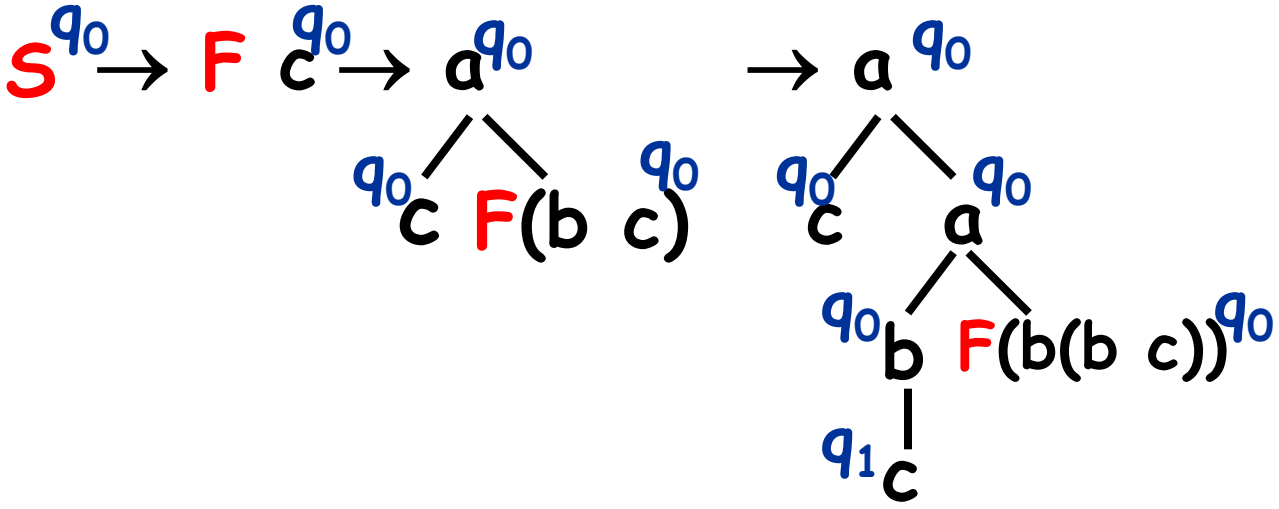
Example

- HORS: $S \to F\ c \qquad F \to \lambda x.a\ x\ (F\ (b\ x))$

  $(S{:}o,\ F{:}\ o \to o)$

- Automaton: $q_0 \to_a q_0\ q_0 \qquad q_0 \to_b q_1 \qquad q_1 \to_b q_1$

  $q_0 \to_c \varepsilon \qquad q_1 \to_c \varepsilon$



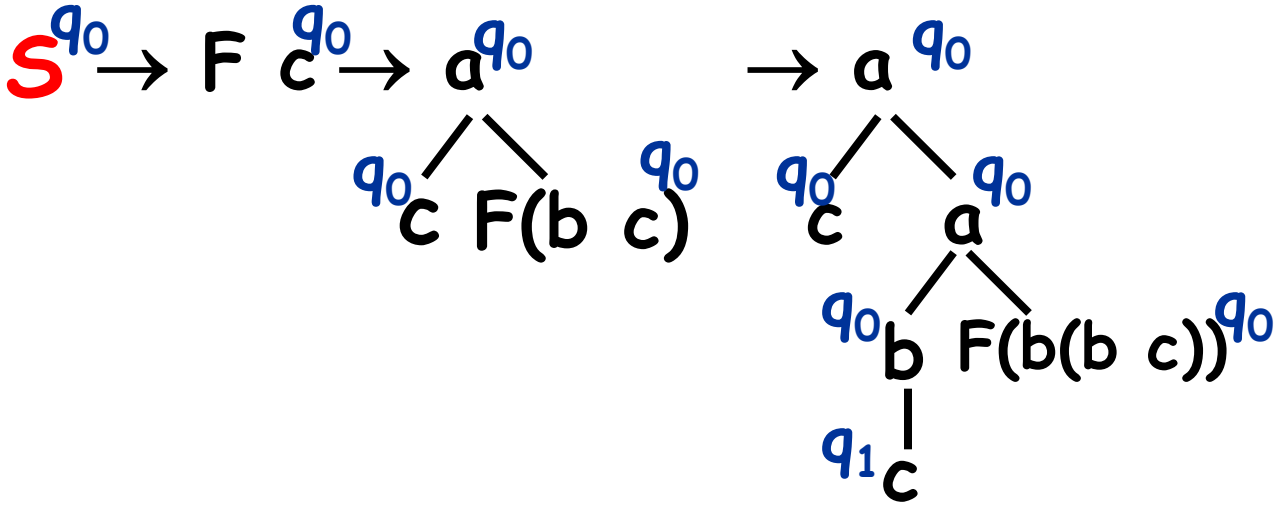
Example

- HORS: $S \to F\ c$ $\quad F \to \lambda x.a\ x\ (F\ (b\ x))$

  $(S{:}o,\ F{:}\ o \to o)$

- Automaton: $q_0 \to_a q_0\ q_0$ $\quad q_0 \to_b q_1$ $\quad q_1 \to_b q_1$

  $q_0 \to_c \varepsilon$ $\quad q_1 \to_c \varepsilon$



Γ_0 :

S: q_0

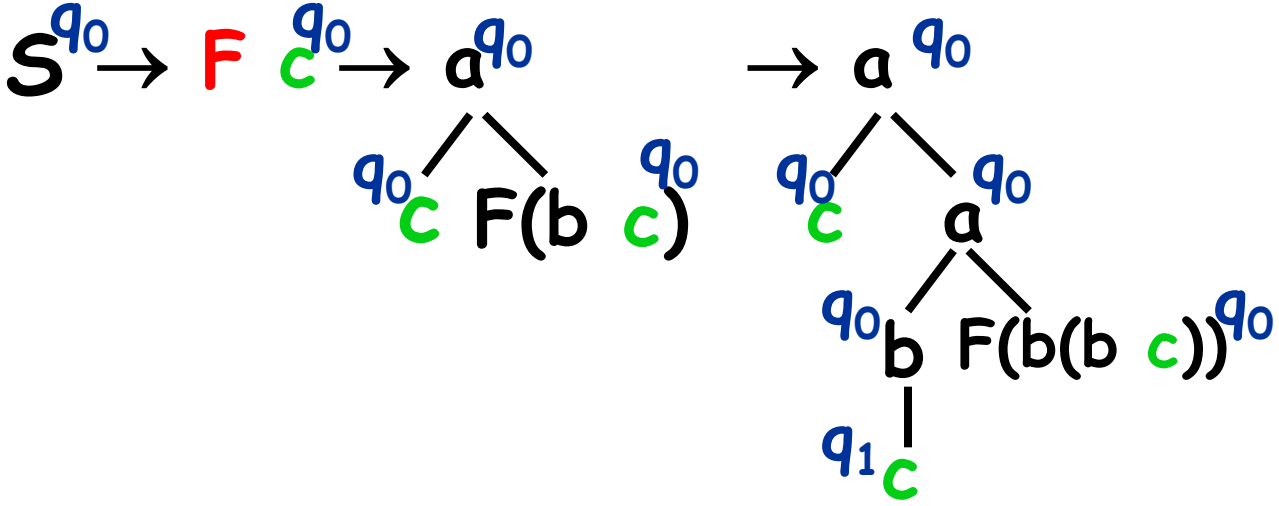
Example

- HORS: $S \to F\ c \qquad F \to \lambda x.a\ x\ (F\ (b\ x))$

  $(S{:}o,\ F{:}\ o \to o)$

- Automaton: $q_0 \to_a q_0\ q_0 \qquad q_0 \to_b q_1 \qquad q_1 \to_b q_1$

  $q_0 \to_c \varepsilon \qquad q_1 \to_c \varepsilon$



Γ_0 :

S: q_0

F: ? → q_0

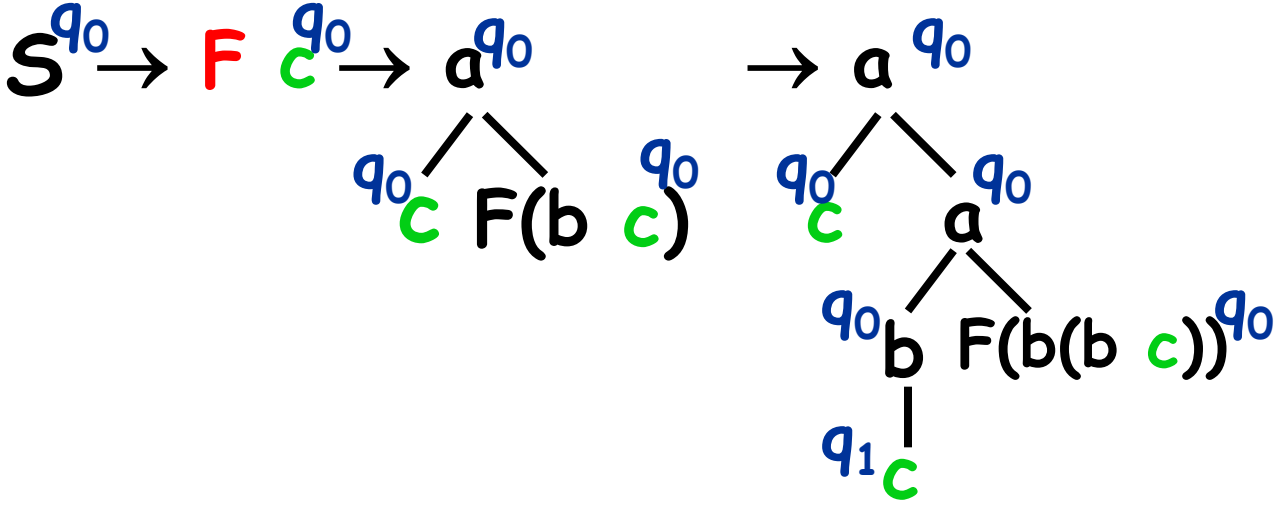
Example

- HORS: $S \to F\ c \qquad F \to \lambda x.a\ x\ (F\ (b\ x))$

  $(S{:}o,\ F{:}\ o \to o)$

- Automaton: $q_0 \to_a q_0\ q_0 \qquad q_0 \to_b q_1 \qquad q_1 \to_b q_1$

  $q_0 \to_c \varepsilon \qquad q_1 \to_c \varepsilon$



Γ_0 :

S: q_0

F: $q_0 \wedge q_1 \to q_0$

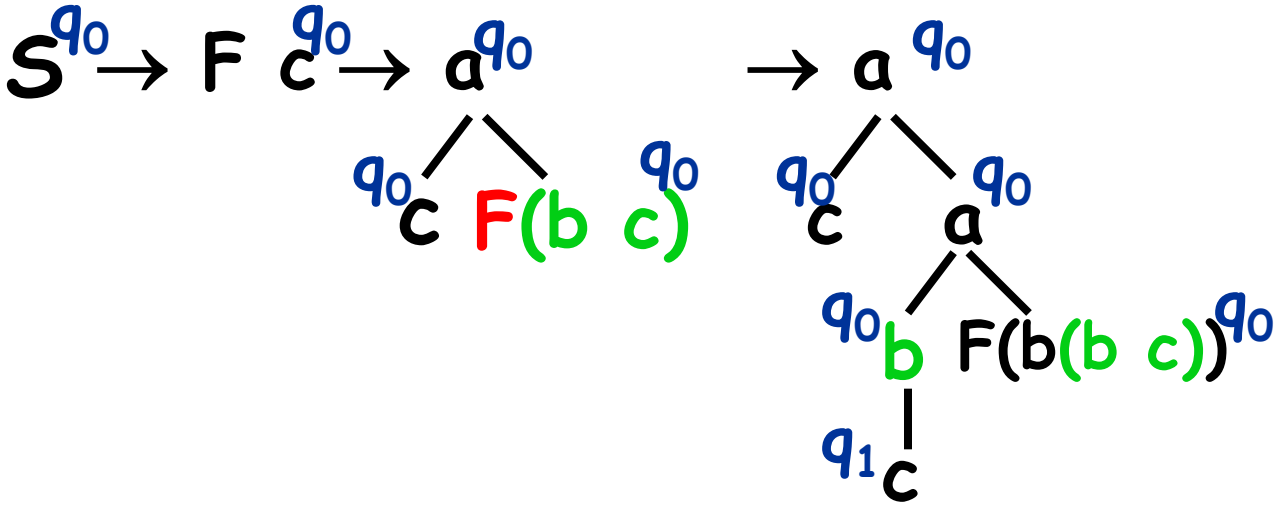
Example

- HORS: $S \to F\ c \qquad F \to \lambda x.a\ x\ (F\ (b\ x))$

  $(S{:}o,\ F{:}\ o \to o)$

- Automaton: $q_0 \to_a q_0\ q_0 \qquad q_0 \to_b q_1 \qquad q_1 \to_b q_1$

  $q_0 \to_c \varepsilon \qquad q_1 \to_c \varepsilon$



Γ_0 :

$S{:}\ q_0$

$F{:}\ q_0 \wedge q_1 \to q_0$

$F{:}\ q_0 \to q_0$

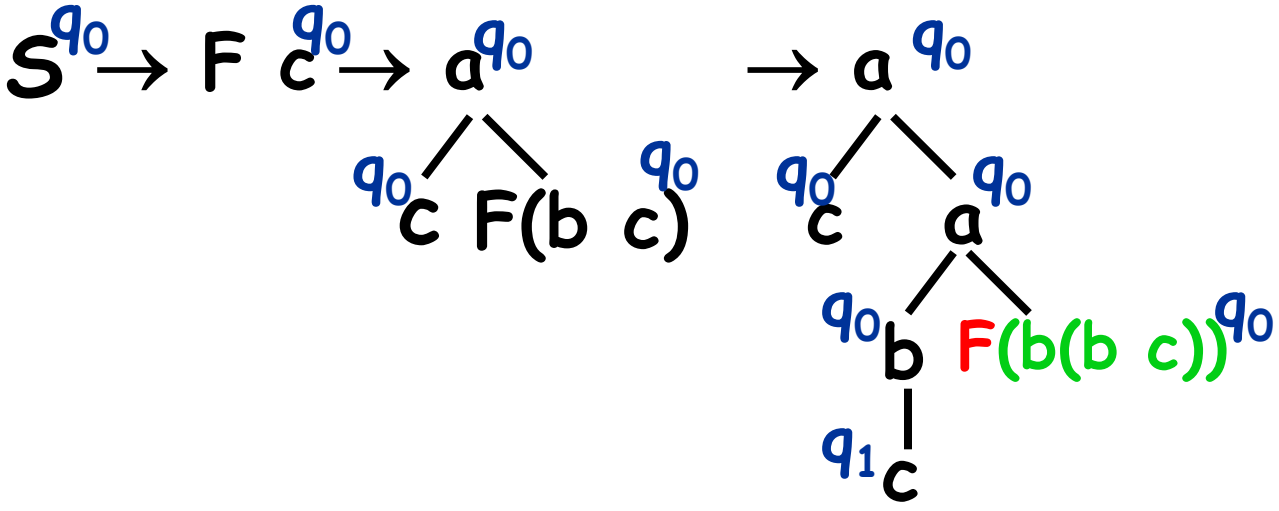
Example

- HORS: $S \to F\ c \qquad F \to \lambda x.a\ x\ (F\ (b\ x))$

  $(S{:}o,\ F{:}\ o \to o)$

- Automaton: $q_0 \to_a q_0\ q_0 \qquad q_0 \to_b q_1 \qquad q_1 \to_b q_1$

  $q_0 \to_c \varepsilon \qquad q_1 \to_c \varepsilon$



Γ_0 :

S: q_0

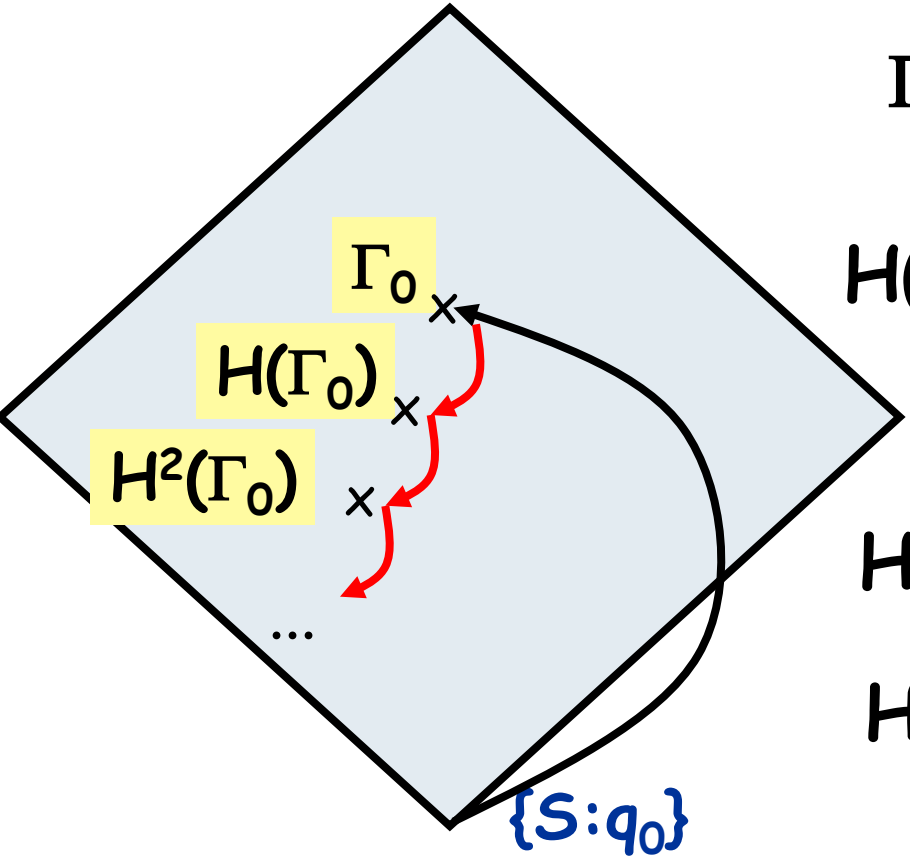
F: $q_0 \wedge q_1 \to q_0$

F: $q_0 \to q_0$

F: $T \to q_0$

Practical Algorithms [K. PPDP09] [K.FoSSaCS11]

1. Guess a type environment Γ_0
2. Compute greatest fixedpoint Γ smaller than Γ_0
3. Check whether $S{:}q_0 \in \Gamma$
4. Repeat 1-3 until the property is proved or refuted.



$$\Gamma_0 = \{S{:}\ q_0,\ F{:}\ q_0 \wedge q_1 \to q_0,\ F{:}\ q_0 \to q_0\ ,\ F{:}\ T \to q_0\}$$

$$H(\Gamma_0) = \{\ F_j{:}\tau \in \Gamma_0 \mid \Gamma_0 \vdash t_j{:}\tau\ \} = \{S{:}\ q_0,\ F{:}\ q_0 \wedge q_1 \to q_0,\ F{:}\ q_0 \to q_0\ \}$$

$$H^2(\Gamma_0) = \{S{:}\ q_0,\ F{:}\ q_0 \wedge q_1 \to q_0\}$$

$$H^3(\Gamma_0) = \{S{:}\ q_0,\ F{:}\ q_0 \wedge q_1 \to q_0\}$$

TRecS [K. PPDP09]

http://www-kb.is.s.u-tokyo.ac.jp/~koba/trecs/

- The first model checker for recursion schemes
- Based on the PPDP09 algorithm, with certain additional optimizations

第2部まとめ

- 高階モデル検査問題は型判定問題に帰着可能
- （帰着で得られる）型判定問題自身もn重指数完全だが、適切な型環境の候補が得られればその妥当性の判定は容易
  （cf. NP完全問題）
- 実践的アルゴリズムでは型環境の候補の求め方を工夫することによって、多くの入力について効率よく動作

第2部に関する参考文献

- 無限木用のオートマトン、時相論理
  - Erich Grädel, Wolfgang Thomas, Thomas Wilke (Eds.): Automata, Logics, and Infinite Games: A Guide to Current Research [outcome of a Dagstuhl seminar, February 2001]. Springer 2002 ISBN 3-540-00388-6
- 高階モデル検査の型検査への帰着
  - Naoki Kobayashi: Model Checking Higher-Order Programs. J. ACM 60(3): 20 (2013)
  - Naoki Kobayashi, C.-H. Luke Ong: A Type System Equivalent to the Modal Mu-Calculus Model Checking of Higher-Order Recursion Schemes. LICS 2009: 179-188
- 最新の高階モデル検査アルゴリズム
  - Christopher Broadbent and Naoki Kobayashi, Saturation-Based Model Checking of Higher-Order Recursion Schemes, CSL 13.
  - Steven J. Ramsay, Robin P. Neatherway, C.-H. Luke Ong, An Abstraction Refinement Approach to Higher-Order Model Checking, HOPA 2013.