Глава 28

Метатеория ограниченной квантификации

В этой главе мы разрабатываем алгоритмы вычисления отношения образования подтипов и проверки типов для $F_{<:}$. Изучается как ядерная, так и полная версия системы; ведут они себя несколько по-разному. Некоторые свойства присутствуют в обоих вариантах, но в полном доказать их сложнее; другие характеристики в полной $F_{<:}$ попросту утрачиваются — такова цена, которую приходится платить за большую выразительность этой системы.

Сначала, в §28.1 и §28.2, мы представим алгоритм проверки типов, работающий в обеих системах. Затем мы рассмотрим проверку образования подтипов, сначала для ядерной системы в §28.3, а потом для полной в §28.4. В §28.5 продолжается обсуждение образования подтипов в полной $F_{<:}$, и особое внимание уделяется тому удивительному обстоятельству, что отношение образования подтипов неразрешимо. В §28.6 мы покажем, что в ядерной $F_{<:}$ имеются пересечения и объединения, а в полной их нет. §28.7 затрагивает некоторые вопросы, связанные с ограниченными экзистенциальными типами, а в §28.8 мы рассмотрим, к чему приводит добавление минимального типа Bot.

28.1. Выявление

В алгоритме проверки типов для простого типизированного лямбда-исчисления с образованием подтипов из §16.2 ключевой идеей было вычислять *минимальный тип* для каждого терма, исходя из минимальных типов его подтермов. Ту же самую базовую идею можно использовать и для $F_{<:}$, однако нужно принять

В этой главе изучается чистая система $F_{<:}$ (рис. 26.1). Соответствующая реализация на OCaml называется purefsub; реализация fullfsub включает также экзистенциальные типы (24.1) и некоторые расширения из главы 11.

$\to \forall <: \mathsf{Top}$

Выявление $\boxed{\Gamma \vdash T \Uparrow T'}$

$$\frac{\mathbf{X<:T} \in \mathbf{\Gamma} \qquad \mathbf{\Gamma \mapsto T \Uparrow T'}}{\mathbf{\Gamma \mapsto X \Uparrow T'}} \quad (\text{XA-Promote})$$

$$\frac{\mathbf{T \text{ не является типовой переменной}}}{\mathbf{\Gamma \mapsto T \Uparrow T}} \quad (\text{XA-Other})$$

Рис. 28.1. Алгоритм выявления для $F_{<:}$

во внимание небольшую сложность, которая возникает из-за наличия в системе типовых переменных. Рассмотрим терм

$$f = \lambda X<:\mathsf{Nat} \to \mathsf{Nat}.\, \lambda y:X.\, y\ 5;$$

$$\triangleright f : \forall X<:\mathsf{Nat} \to \mathsf{Nat}.\, X \to \mathsf{Nat}$$

Ясно, что этот терм правильно типизирован, поскольку тип переменной y в применении $y\ 5$ может быть расширен до $\mathsf{Nat}\to\mathsf{Nat}$ по правилу T-Sub. Однако *минимальный* тип y равен X, и это не функциональный тип. Чтобы определить минимальный тип всего терма-применения, нужно найти наименьший функциональный тип для y — т. е. минимальный функциональный тип, являющийся надтипом типовой переменной X. Неудивительно, что такой тип можно найти, *расширяя* минимальный тип y, пока он не превратится в нечто отличное от типовой переменной.

Мы используем формальную запись $\Gamma \vdash S \Uparrow T$ (произносится как «S выявляется (exposes) как T в контексте Γ»), что означает «T — минимальный надтип S, не являющийся переменной». Выявление определяется через циклическое расширение типовых переменных, как показано на рис. 28.1.

Нетрудно убедиться, что эти правила дают всюду определенную функцию. Более того, результатом выявления типа всегда будет наименьший надтип, отличный от переменной. Например, если $\Gamma = X<:\mathsf{Top}, Y<:\mathsf{Nat}\boxtimes\mathsf{Nat}, Z<:Y, W<:Z$, то

$$\Gamma \vdash \mathsf{Top} \Uparrow \mathsf{Top} \quad \Gamma \vdash Y \Uparrow \mathsf{Nat}\to\mathsf{Nat} \quad \Gamma \vdash W \Uparrow \mathsf{Nat}\to\mathsf{Nat}$$
$$\Gamma \vdash X \Uparrow \mathsf{Top} \qquad \Gamma \vdash Z \Uparrow \mathsf{Nat}\to\mathsf{Nat}$$

Основные свойства выявления можно описать следующим образом.

Лемма 28.1.1 (Выявление)**.** Допустим, $\Gamma \vdash S \Uparrow T$. Тогда

1. $\Gamma \vdash S <: T$
2. Если $\Gamma \vdash S <: U$, где U — не переменная, то $\Gamma \vdash T <: U$.

Доказательство. Часть (1) доказывается индукцией по деревьям вывода $\Gamma \vdash S \Uparrow T$, часть (2) индукцией по деревьям вывода $\Gamma \vdash S <: U$. □

$\to \forall <: \mathsf{Top}$ *Расширяет* $\lambda_{<:}$ *(16.3)*

Алгоритмическая типизация $\boxed{\Gamma \mapsto t : T}$

$$\frac{x{:}T \in \Gamma}{\Gamma \mapsto x : T} \quad \text{(TA-Var)}$$

$$\frac{\Gamma, x{:}T_1 \mapsto t_2 : T_2}{\Gamma \mapsto \lambda x{:}T_1 . t_2 : T_1 \to T_2} \quad \text{(TA-Abs)}$$

$$\frac{\Gamma \mapsto t_1 : T_1 \qquad \mathbf{\Gamma} \mapsto T_1 \Uparrow (T_{11} \to T_{12}) \qquad \Gamma \mapsto t_2 : T_2 \qquad \mathbf{\Gamma} \mapsto T_2 <: T_{11}}{\Gamma \mapsto t_1\, t_2 : T_{12}} \quad \text{(TA-App)}$$

$$\frac{\mathbf{\Gamma, X{<:}T_1 \mapsto t_2 : T_2}}{\mathbf{\Gamma \mapsto \lambda X{<:}T_1 . t_2 : \forall X{<:}T_1 . T_2}} \quad \text{(TA-TAbs)}$$

$$\frac{\mathbf{\Gamma \mapsto t_1 : T_1 \qquad \Gamma \mapsto T_1 \Uparrow \forall X{<:}T_{11} . T_{12} \qquad \Gamma \mapsto T_2 <: T_{11}}}{\mathbf{\Gamma \mapsto t_1\,[T_2] : [X \mapsto T_2]T_{12}}} \quad \text{(TA-TApp)}$$

Рис. 28.2. Алгоритмическая типизация для $F_{<:}$

28.2. Минимальная типизация

Алгоритм вычисления минимальных типов строится по тем же принципам, что и для простого типизированного лямбда-исчисления с образованием подтипов, но с одной дополнительной деталью: когда мы проверяем тип в терме-применении, мы сначала вычисляем минимальный тип левой части, а затем *выявляем* его, получая функциональный тип, как показано на рис. 28.2. Если же при выявлении левой части применения не получается функционального типа, то правило TA-App оказывается неприменимо, и терм типизирован неверно. Аналогично мы проверяем типы в применении типа: выявляем левую часть и надеемся при этом получить кванторный тип.

Доказательство корректности и полноты этого алгоритма по отношению к исходным правилам типизации не представляют труда. Мы приводим доказательство для ядерной $F_{<:}$ (рассуждение для полной $F_{<:}$ строится аналогично; ср. упражнение 28.2.3).

Теорема 28.2.1 (Минимальная типизация)**.**

1. Если $\Gamma \mapsto t : T$, то $\Gamma \vdash t : T$.

2. Если $\Gamma \vdash t : T$, то $\Gamma \mapsto t : M$, причем $\Gamma \vdash M <: T$.

Доказательство. Часть (1) представляет собой несложную индукцию по алгоритмическим выводам, с использованием части (1) леммы 28.1.1 для вариантов с применениями. В части (2) проводится индукция по дереву вывода $\Gamma \vdash t : T$, с разбором вариантов последнего правила в выводе. Наиболее интересны варианты T-App и T-TApp.

Вариант T-Var: $t = x \quad x{:}T \in \Gamma$
По правилу TA-Var, $\Gamma \mapsto x : T$. По правилу S-Refl, $\Gamma \vdash T <: T$.

Вариант T-Abs: $t = \lambda x{:}T_1.t_2 \quad \Gamma, x{:}T_1 \vdash t_2 : T_2 \quad T = T_1 \to T_2$
Согласно предположению индукции, $\Gamma, x{:}T_1 \mapsto t_2 : M_2$ для некоторого M_2, причем $\Gamma, x{:}T_1 \vdash M_2 <: T_2$ — т. е. $\Gamma \vdash M_2 <: T_2$, поскольку отношение образования подтипов не зависит от связываний термовых переменных в контексте (лемма 26.4.4). По правилу TA-Abs, $\Gamma \mapsto t : T_1 \to M_2$. По правилам S-Refl и S-Arrow, $\Gamma \vdash T_1 \to M_2 <: T_1 \to T_2$.

Вариант T-App: $t = t_1\, t_2 \quad \Gamma \vdash t_1 : T_{11} \to T_{12} \quad T = T_{12} \quad \Gamma \vdash t_2 : T_{11}$
Согласно предположению индукции, имеем $\Gamma \mapsto t_1 : M_1$ и $\Gamma \mapsto t_2 : M_2$, причем $\Gamma \vdash M_1 <: T_{11} \to T_{12}$ и $\Gamma \vdash M_2 <: T_{11}$. Пусть N_1 — наименьший надтип M_1, не являющийся переменной, т. е. $\Gamma \vdash M_1 \Uparrow N_1$. Согласно части (2) леммы 28.1.1, $\Gamma \vdash N_1 <: T_{11} \to T_{12}$. Поскольку мы знаем, что N_1 — не переменная, лемма об инверсии для отношения образования подтипов (26.4.10) сообщает нам, что $N_1 = N_{11} \to N_{12}$, причем $\Gamma \vdash T_{11} <: N_{11}$ и $\Gamma \vdash N_{12} <: T_{12}$. По транзитивности, $\Gamma \vdash M_2 <: N_{11}$, так что применимо правило TA-App, и оно дает нам $\Gamma \mapsto t_1\, t_2 : N_{12}$. Все требования теоремы при этом оказываются соблюдены.

Вариант T-TAbs: $t = \lambda X{<:}T_1.t_2 \quad \Gamma, X{<:}T_1 \vdash t_2 : T_2 \quad T = \forall X{<:}T_1.T_2$
По предположению индукции, $\Gamma, X{<:}T_1 \mapsto t_2 : M_2$ для некоторого M_2, причем $\Gamma, X{<:}T_1 \vdash M_2 <: T_2$. По правилу TA-TAbs, $\Gamma \mapsto t : \forall X{<:}T_1.M_2$. По правилу S-All, $\Gamma \vdash \forall X{<:}T_1.M_2 <: \forall X{<:}T_1.T_2$.

Вариант T-TApp: $t = t_1\,[T_2] \quad T = [X \mapsto T_2]T_{12} \quad \Gamma \vdash t_1 : \forall X{<:}T_{11}.T_{12} \quad \Gamma \vdash T_2 <: T_{11}$
Согласно предположению индукции, имеем $\Gamma \mapsto t_1 : M_1$, причем $\Gamma \vdash M_1 <: \forall X{<:}T_{11}.T_{12}$. Пусть N_1 — наименьший надтип M_1, не являющийся переменной, т. е. $\Gamma \vdash M_1 \Uparrow N_1$. По лемме о выявлении (28.1.1), $\Gamma \vdash N_1 <: \forall X{<:}T_{11}.T_{12}$. Но мы знаем, что N_1 — не переменная, так что по лемме об инверсии для отношения образования подтипов (26.4.10), имеем $N_1 = \forall X{<:}T_{11}.N_{12}$, причем $\Gamma, X{<:}T_{11} \vdash N_{12} <: T_{12}$. Правило TA-TApp дает нам $\Gamma \mapsto t_1\,[T_2] : [X \mapsto T_2]N_{12}$, а поскольку отношение образования подтипов сохраняется при подстановке (лемма 26.4.8), $\Gamma \vdash [X \mapsto T_2]N_{12} <: [X \mapsto T_2]T_{12} = T$.

Вариант T-Sub: $\Gamma \vdash t : S \quad \Gamma \vdash S <: T$
Согласно предположению индукции, $\Gamma \mapsto t : M$, причем $\Gamma \vdash M <: S$. По транзитивности, $\Gamma \vdash M <: T$. □

Следствие 28.2.2 (Разрешимость типизации)**.** Отношение типизации для ядерной $F_{<:}$ разрешимо, если имеется разрешающая процедура для отношения образования подтипов.

Доказательство. Для любых Γ и t можно проверить, существует ли какой-либо тип T, такой что $\Gamma \vdash t : T$, породив при помощи алгоритмических правил доказательство $\Gamma \mapsto t : T$. Если доказательство прошло успешно, то полученный T является также типом для t согласно исходному отношению типизации, по части (1) теоремы 28.2.1. В противном случае, из части (2) теоремы 28.2.1 следует, что t не

имеет типа в исходном отношении типизации. Наконец, заметим, что алгоритмические правила типизации соответствуют всегда завершающемуся алгоритму, поскольку они управляются синтаксисом (к каждому данному терму t применимо не более одного правила) и всегда уменьшают размер t в направлении снизу вверх. □

Упражнение 28.2.3 (⋆⋆). Как нужно изменить вышеприведенное доказательство, чтобы оно работало с полной $F_{<:}$?

28.3. Образование подтипов в ядерной $F_{<:}$

В §16.1 мы отмечали, что декларативное отношение образования подтипов для простого типизированного лямбда-исчисления с образованием подтипов не управляется синтаксисом, т. е. его невозможно прочесть как алгоритм образования подтипов, по двум причинам: (1) заключения правил S-Refl и S-Trans перекрываются с другими правилами (так что, если мы читаем правила снизу вверх, мы не знаем, которое из правил следует применить), и (2) в предпосылках S-Trans упоминается метапеременная, которая не встречается в его заключении (и наивному алгоритму пришлось бы как-то «угадывать» ее значение). Мы видели, что можно справиться с этими проблемами, просто исключив неудобные для алгоритмизации правила из системы. Однако для этого нам сначала потребовалось немного подправить систему, сведя три раздельных правила образования подтипов для записей в одно.

Для ядерной $F_{<:}$ ситуация аналогична. Неудобными правилами снова являются S-Refl и S-Trans, и мы получаем алгоритм, устраняя эти правила и изменяя остающиеся правила так, что в результате они берут на себя обработку тех случаев, которые раньше обрабатывались устраненными правилами.

В простом типизированном лямбда-исчислении с образованием подтипов не было случаев, которые требовали бы наличия правила рефлексивности — его можно было просто отбросить, не повлияв на множество выводимых утверждений об образовании подтипов (лемма 16.1.2, часть 1). Напротив, в $F_{<:}$ утверждения об образовании подтипов вида $\Gamma \vdash X <: X$ доказываются только через рефлексивность. Так что, когда мы удаляем полное правило рефлексивности, вместо него следует добавить ограниченную аксиому рефлексивности, касающуюся только переменных.

$$\Gamma \vdash X <: X$$

Аналогично, чтобы избавиться от правила S-Trans, следует сначала понять, какие случаи его использования неустранимы. Здесь интерес представляет взаимодействие с правилом S-TVar, которое позволяет использовать предположения о типовых переменных при выводе утверждений об образовании подтипов. Например, если $\Gamma = W<:\mathsf{Top}, X<:W, Y<:X, Z<:Y$, то утверждение $\Gamma \vdash Z <: W$ невозможно вывести, если в системе отсутствует правило S-Trans. В общем случае экземпляр S-Trans, где

$\to \forall <: \mathsf{Top}$ *Расширяет* $\lambda_{<:}$ *(16.2)*

Алгоритмическое образование подтипов $\boxed{\Gamma \mapsto S <: T}$

$$\frac{}{\Gamma \mapsto S <: \mathsf{Top}} \text{ (SA-Top)}$$

$$\frac{}{\Gamma \mapsto X <: X} \text{ (SA-Refl-TVar)}$$

$$\frac{X<:U \in \Gamma \qquad \Gamma \mapsto U <: T}{\Gamma \mapsto X <: T} \text{ (SA-Trans-TVar)}$$

$$\frac{\Gamma \mapsto T_1 <: S_1 \qquad \Gamma \mapsto S_2 <: T_2}{\Gamma \mapsto S_1 \to S_2 <: T_1 \to T_2} \text{ (SA-Arrow)}$$

$$\frac{\Gamma, X<:U_1 \mapsto S_2 <: T_2}{\Gamma \mapsto \forall X<:U_1.S_2 <: \forall X<:U_1.T_2} \text{ (SA-All)}$$

Рис. 28.3. Алгоритмическое отношение образования подтипов для ядерной $F_{<:}$

левый подвывод является экземпляром аксиомы S-TVar, как в

$$\frac{\dfrac{Z <: Y \in \Gamma}{\Gamma \vdash Z <: Y}\text{ (S-TVar)} \qquad \dfrac{\vdots}{\Gamma \vdash Y <: W}}{\Gamma \vdash Z <: W}\text{ (S-Trans)}$$

неустраним.

К счастью, выводы такого вида — *единственный* важный неустранимый класс случаев использования транзитивности при выводе образования подтипов. Это наблюдение можно выразить точно, введя новое правило образования подтипов

$$\frac{X<:U \in \Gamma \qquad \Gamma \vdash U <: T}{\Gamma \vdash X <: T}$$

которое охватывает в точности эту форму поиска переменной, за которым следует применение транзитивности, и показывает, что замена правил транзитивности и поиска переменной этим правилом не влияет на множество выводимых утверждений об образовании подтипов.

Эти изменения приводят нас к алгоритмическому отношению образования подтипов для ядерной $F_{<:}$, изображенному на рис. 28.3. Мы добавляем стрелку к концу символа «штопора» в алгоритмических утверждениях о типизации, чтобы отличить их от исходной формы утверждений о типизации, когда речь в тексте идет об обеих разновидностях.

Тот факт, что новых правил SA-Refl-TVar и SA-Trans-TVar достаточно для замены старых правил рефлексивности и транзитивности, подтверждается следующими двумя леммами.

Лемма 28.3.1 (Рефлексивность алгоритмического отношения образования подтипов)**.** Для каждого типа T и контекста Γ можно доказать, что $\Gamma \mapsto T <: T$.

Доказательство. Индукция по T. □

Лемма 28.3.2 (Транзитивность алгоритмического отношения образования подтипов). Если $\Gamma \mapsto S <: Q$ и $\Gamma \mapsto Q <: T$, то $\Gamma \mapsto S <: T$.

Доказательство. Индукция по сумме размеров двух деревьев вывода. Имея два подвывода, мы рассматриваем последние правила обоих этих подвыводов.

Если правый подвывод является экземпляром SA-Top, то доказательство закончено, поскольку $\Gamma \mapsto S <: \mathrm{Top}$ по правилу SA-Top. Если левый подвывод — экземпляр SA-Top, то $Q = \mathrm{Top}$, и, рассматривая алгоритмические правила, мы видим, что и правый подвывод обязан быть экземпляром SA-Top.

Если какой-либо подвывод является экземпляром SA-Refl-TVar, то, опять же, все доказано, поскольку другое поддерево будет в точности являться желаемым результатом.

Если левый подвывод завершается экземпляром SA-Trans-TVar, то $S = Y$, причем $Y<:U \in \Gamma$, и у нас есть подвывод с заключением $\Gamma \mapsto U <: Q$. Согласно предположению индукции, $\Gamma \mapsto U <: T$, и, снова по правилу SA-Trans-TVar, имеем $\Gamma \mapsto Y <: T$, что и требуется.

Если левое поддерево заканчивается экземпляром SA-Arrow, имеем $S = S_1 \to S_2$ и $Q = Q_1 \to Q_2$, с подвыводами $\Gamma \mapsto Q_1 <: S_1$ и $\Gamma \mapsto S_2 <: Q_2$. Однако, поскольку мы уже рассмотрели вариант, в котором правый подвывод представляет собой SA-Top, единственная оставшаяся возможность состоит в том, что этот вывод также заканчивается на SA-Arrow, а значит, имеем $T = T_1 \to T_2$ и еще два подвывода $\Gamma \mapsto T_1 <: Q_1$ и $\Gamma \mapsto Q_2 <: T_2$. Теперь дважды применяем предположение индукции, получая $\Gamma \mapsto T_1 <: S_1$ и $\Gamma \mapsto S_2 <: T_2$. Наконец, SA-Arrow дает нам $\Gamma \mapsto S_1 \to S_2 <: T_1 \to T_2$, что и требуется.

В случае, когда левый подвывод заканчивается экземпляром SA-All, рассуждение проходит аналогично. Имеем $S = \forall X<:U_1.S_2$ и $Q = \forall X<:U_1.Q_2$, а также подвывод $\Gamma, X<:U_1 \mapsto S_2 <: Q_2$. Опять же, поскольку мы уже рассмотрели вариант, когда правый подвывод является экземпляром SA-Top, он должен заканчиваться на SA-All; так что $T = \forall X<:U_1.T_2$, причем имеется подвывод $\Gamma, X<:U_1 \mapsto Q_2 <: T_2$. Из предположения индукции получаем $\Gamma, X<:U_1 \mapsto S_2 <: T_2$, и, по правилу SA-All, $\Gamma \mapsto \forall X<:U_1.S_2 <: \forall X<:U_1.T_2$. □

Теорема 28.3.3 (Корректность и полнота алгоритмического отношения образования подтипов). $\Gamma \vdash S <: T$ тогда и только тогда, когда $\Gamma \mapsto S <: T$.

Доказательство. В обоих направлениях проводится индукция по деревьям вывода. Корректность ($\Leftarrow$) не представляет труда. При доказательстве полноты ($\Rightarrow$) используются леммы 28.3.1 и 28.3.2. □

Наконец, требуется убедиться в том, что правила для образования подтипов определяют *тотальный* алгоритм — т. е. алгоритм, завершающийся при любых входных данных. Мы сделаем это, присваивая каждому утверждению об образовании подтипов вес, и показывая, что каждое алгоритмическое правило имеет заключение со строго большим весом, чем у предпосылок.

$\to \forall <: \mathsf{Top}$ ***полная*** *Расширяет 28.3*

Алгоритмическое образование подтипов $\boxed{\Gamma \mapsto S <: T}$

$$\frac{}{\Gamma \mapsto S <: \mathsf{Top}} \quad \text{(SA-Top)}$$

$$\frac{}{\Gamma \mapsto X <: X} \quad \text{(SA-Refl-TVar)}$$

$$\frac{X<:U \in \Gamma \qquad \Gamma \mapsto U <: T}{\Gamma \mapsto X <: T} \quad \text{(SA-Trans-TVar)}$$

$$\frac{\Gamma \mapsto T_1 <: S_1 \qquad \Gamma \mapsto S_2 <: T_2}{\Gamma \mapsto S_1 \to S_2 <: T_1 \to T_2} \quad \text{(SA-Arrow)}$$

$$\frac{\boldsymbol{\Gamma \mapsto T_1 <: S_1} \qquad \Gamma, X<:\boldsymbol{T_1} \mapsto S_2 <: T_2}{\Gamma \mapsto \forall X<:\boldsymbol{S_1}.S_2 <: \forall X<:\boldsymbol{T_1}.T_2} \quad \text{(SA-All)}$$

Рис. 28.4. Алгоритмическое отношение образования подтипов для полной $F_{<:}$

Определение 28.3.4. *Вес* типа T в контексте Γ, который обозначается как $\mathit{weight}_\Gamma(T)$, определяется так:

$$\begin{aligned}
\mathit{weight}_\Gamma(X) &= \mathit{weight}_\Gamma(U) + 1 \quad \text{если } \Gamma = \Gamma_1, X <: U, \Gamma_2 \\
\mathit{weight}_\Gamma(\mathsf{Top}) &= 1 \\
\mathit{weight}_\Gamma(T_1 \to T_2) &= \mathit{weight}_\Gamma(T_1) + \mathit{weight}_\Gamma(T_2) + 1 \\
\mathit{weight}_\Gamma(\forall X<:T_1.T_2) &= \mathit{weight}_{\Gamma, X<:T_1}(T_2) + 1
\end{aligned}$$

Вес утверждения об образовании подтипов $\Gamma \vdash S <: T$ есть сумма весов S и T в контексте Γ.

Теорема 28.3.5. Алгоритм проверки образования подтипов завершается при любом вводе.

Доказательство. Вес заключения в любом экземпляре алгоритмического правила образования подтипов всегда строго больше, чем вес каждой из предпосылок. □

Следствие 28.3.6. Отношение образования подтипов в ядерной $F_{<:}$ разрешимо.

28.4. Образование подтипов в полной $F_{<:}$

Алгоритм проверки образования подтипов для полной $F_{<:}$, приведенный на рис. 28.4, почти такой же, как для ядерной $F_{<:}$; единственное изменение заключается в замене SA-All на более гибкий вариант. Как и в случае ядерной $F_{<:}$, корректность и полнота этого алгоритмического отношения по сравнению с исходным отношением образования подтипов прямо следуют из рефлексивности и транзитивности алгоритмического отношения.

Рассуждение для рефлексивности остается в точности таким же, как раньше, однако доказательство транзитивности оказывается несколько более тонким. Чтобы понять, почему это так, вспомним доказательство транзитивности для ядерной $F_{<:}$ из предыдущего раздела (лемма 28.3.2). Там идея состояла в том, чтобы взять два дерева вывода для отношения образования подтипов, завершающиеся утверждениями $\Gamma \vdash S <: Q$ и $\Gamma \vdash Q <: T$, и показать, как переставить и пересобрать их поддеревья, получая при этом вывод $\Gamma \vdash S <: T$, не используя правило транзитивности и предполагая (в качестве предположения индукции), что то же самое можно проделать для выводов меньшего размера. Предположим теперь, что у нас есть два подвывода, завершающиеся новым правилом SA-All:

$$\frac{\dfrac{\vdots}{\Gamma \vdash Q_1 <: S_1} \qquad \dfrac{\vdots}{\Gamma, X<:Q_1 \vdash S_2 <: Q_2}}{\Gamma \vdash \forall X<:S_1.S_2 <: \forall X<:Q_1.Q_2} \qquad \frac{\dfrac{\vdots}{\Gamma \vdash T_1 <: Q_1} \qquad \dfrac{\vdots}{\Gamma, X<:T_1 \vdash Q_2 <: T_2}}{\Gamma \vdash \forall X<:Q_1.Q_2 <: \forall X<:T_1.T_2}$$

Следуя схеме предыдущего доказательства, мы хотели бы воспользоваться предположением индукции, чтобы объединить левый и правый подвыводы и получить единственный экземпляр SA-All с заключением $\Gamma \vdash \forall X<:S_1.S_2 <: \forall X<:T_1.T_2$. Для левых подвыводов никаких сложностей нет; предположение индукции дает нам вывод $\Gamma \vdash T_1 <: S_1$, не использующий транзитивности. Однако для правых подвыводов предположение индукции неприменимо, поскольку контексты подвыводов различаются: верхняя граница для X в одном из них равна Q_1, а в другом — T_1.

К счастью, мы знаем, как *сделать* контексты одинаковыми: свойство *сужения* из главы 26 (лемма 26.4.5) говорит, что истинное утверждение об образовании подтипов остается истинным, если мы заменяем граничный тип в контексте одним из его подтипов. Так что, казалось бы, можно просто сузить подвывод $\Gamma, X<:Q_1 \vdash S_2 <: Q_2$ до $\Gamma, X<:T_1 \vdash S_2 <: Q_2$, и таким образом разрешить использование предположения индукции.

Тут, однако, необходима известная осторожность. Лемма 26.4.5 говорит, что можно взять произвольный вывод и породить вывод с суженным заключением, однако она *не гарантирует*, что размер нового вывода будет таким же, как у старого. В самом деле, рассмотрев доказательство этой леммы, мы увидим, что, как правило, сужение порождает вывод большего размера, чем исходный, поскольку всюду, где аксиома S-TVar используется для поиска сужаемой переменной, вставляется копия произвольно большого вывода. Более того, эта операция вставки приводит к порождению новых экземпляров правила транзитивности, а именно отсутствие необходимости в этом правиле в нашей текущей системе мы пытаемся доказать.

Чтобы справиться с этими сложностями, мы доказываем транзитивность и сужение *совместно*, с предположением индукции, основанном на размере промежуточного типа Q для свойства транзитивности и размере исходного типа-границы Q для свойства сужения.

Прежде чем начать основное доказательство, мы приводим несложную лемму, утверждающую, что порядок добавления новых связываний типовых пере-

менных в контекст не влияет на верность выводимых утверждений об образовании подтипов.

Лемма 28.4.1 (Перестановка и ослабление)**.**

1. Предположим, что Δ является правильно сформированной перестановкой Γ (ср. 26.4.1). Если $\Gamma \vdash S <: T$, то $\Delta \vdash S <: T$.

2. Если $\Gamma \vdash S <: T$ и $dom(\Delta) \cap dom(\Gamma) = \varnothing$, то $\Gamma, \Delta \vdash S <: T$.

Доказательство. Прямолинейное доказательство по индукции. Часть (1) используется в варианте SA-All части (2). □

Лемма 28.4.2 (Транзитивность и сужение для полной $F_{<:}$)**.**

1. Если $\Gamma \vdash S <: Q$ и $\Gamma \vdash Q <: T$, то $\Gamma \vdash S <: T$.

2. Если $\Gamma, X<:Q, \Delta \vdash M <: N$ и $\Gamma \vdash P <: Q$, то $\Gamma, X<:P, \Delta \vdash M <: N$.

Доказательство. Обе части леммы доказываются одновременно, индукцией по размеру Q. На каждом шаге индукции в доказательстве части (2) мы предполагаем, что часть (1) уже доказана для нашего Q; часть (1) использует часть (2) только для типов Q строго меньшего размера.

1. Проводим внутреннюю индукцию по размеру первого данного вывода, и анализируем последнее правило в обоих подвыводах. Все варианты, кроме одного, совпадают с доказательством леммы 28.3.2; различие касается случая SA-All.

Если правый подвывод является экземпляром SA-Top, то доказательство закончено, поскольку $\Gamma \vdash S <: \mathrm{Top}$ по правилу SA-Top. Если левый подвывод — экземпляр SA-Top, то $Q = \mathrm{Top}$ и, рассматривая алгоритмические правила, мы видим, что и правый подвывод тогда обязан быть экземпляром SA-Top. Если какой-либо из подвыводов является экземпляром SA-Refl-TVar, то, опять же, лемма доказана, поскольку другое поддерево будет в точности желаемым результатом.

Если левое поддерево заканчивается экземпляром правила SA-Trans-TVar, то имеем $S = Y$, причем $Y<:U \in \Gamma$, и это выражение есть подвывод утверждения $\Gamma \vdash U : Q$. По внутреннему предположению индукции, $\Gamma \vdash U <: T$, и, снова по SA-Trans-TVar, $\Gamma \vdash Y <: T$, что нам и требуется.

Если левое поддерево заканчивается экземпляром правил SA-Arrow или SA-All, то, поскольку вариант с правилом SA-Top в качестве правого подвывода уже рассмотрен, правый подвывод должен завершаться тем же правилом, что и левый. Если это правило — SA-Arrow, то имеем $S = S_1 \to S_2$, $Q = Q_1 \to Q_2$ и $T = T_1 \to T_2$ с подвыводами $\Gamma \vdash Q_1 <: S_1$, $\Gamma \vdash S_2 <: Q_2$, $\Gamma \vdash T_1 <: Q_1$ и $\Gamma \vdash Q_2 <: T_2$. Применяем часть (1) внешнего предположения индукции дважды (поскольку как Q_1, так и Q_2 меньше по размеру, чем Q) и получаем $\Gamma \vdash T_1 <: S_1$ и $\Gamma \vdash S_2 <: T_2$. Наконец, с помощью правила SA-Arrow получаем $\Gamma \vdash S_1 \to S_2 <: T_1 \to T_2$.

В случае, когда оба подвывода заканчиваются на SA-All, имеем $S = \forall X<:S_1.S_2$, $Q = \forall X<:Q_1.Q_2$ и $T = \forall X<:T_1.T_2$, и есть подвыводы

$$\Gamma \vdash Q_1 <: S_1 \quad \Gamma, X<:Q_1 \vdash S_2 <: Q_2$$
$$\Gamma \vdash T_1 <: Q_1 \quad \Gamma, X<:T_1 \vdash Q_2 <: T_2$$

Согласно части (1) внешнего предположения индукции (поскольку Q_1 меньше по размеру, чем Q), мы можем объединить два подвывода для ограничений и получить $\Gamma \vdash T_1 <: S_1$. Для тел кванторов приходится приложить немного больше усилий, поскольку контексты не совсем совпадают. Используем часть (2) внешнего предположения индукции (поскольку Q_2 меньше, чем Q) и сужаем ограничение для X в подвыводе $\Gamma, X<:Q_1 \vdash S_2 <: Q_2$, так что получается $\Gamma, X<:T_1 \vdash S_2 <: Q_2$. Теперь применима часть (1) внешнего предположения индукции (поскольку Q_2 меньше, чем Q); она дает нам $\Gamma, X<:T_1 \vdash S_2 <: T_2$. Наконец, по правилу SA-All, $\Gamma \vdash \forall X<:S_1.S_2 <: \forall X<:T_1.T_2$.

2. Снова проведем внутреннюю индукцию по размеру первого данного подвывода, рассматривая варианты последнего правила в нем. В большинстве вариантов нужно всего лишь очевидным образом использовать внутреннее предположение индукции. Интерес представляет вариант SA-Trans-TVar с $M = X$, где в качестве подвывода мы имеем $\Gamma, X<:Q, \Delta \vdash Q <: N$. Применяя внутреннее предположение индукции к этому подвыводу, получаем $\Gamma, X<:P, \Delta \vdash Q <: N$. Кроме того, через ослабление (лемма 28.4.1, часть 2) второго данного вывода получаем $\Gamma, X<:P, \Delta \vdash P <: Q$. Теперь через часть (1) внешнего предположения индукции (с тем же самым Q) имеем $\Gamma, X<:P, \Delta \vdash P <: N$. Наконец, применяем правило SA-Trans-TVar и получаем $\Gamma, X<:P, \Delta \vdash X <: N$, что и требуется. □

Упражнение 28.4.3 (★★★★ ↛). Есть еще один осмысленный вариант правила образования подтипов для кванторов, несколько более гибкий, чем правило ядерной $F_{<:}$, но существенно более слабый, чем правило полной $F_{<:}$:

$$\frac{\Gamma \vdash S_1 <: T_1 \qquad \Gamma \vdash T_1 <: S_1 \qquad \Gamma, X<:T_1 \vdash S_2 <: T_2}{\Gamma \vdash \forall X<:S_1.S_2 <: \forall X<:T_1.T_2} \tag{S-All}$$

Это правило близко к варианту ядерной $F_{<:}$, но требует не синтаксического совпадения границ двух кванторов, а только их эквивалентности — каждый из них должен быть подтипом другого. Разница между ядерным правилом и этим проявляется только тогда, когда мы обогащаем язык какой-нибудь конструкцией, правила образования подтипов которой порождают нетривиальные классы эквивалентности между типами, например, записями. Скажем, в чистой ядерной $F_{<:}$ с записями тип $\forall X<:\{a:Top,b:Top\}.X$ не будет подтипом $\forall X<:\{b:Top,a:Top\}.X$, а в системе с предлагаемым правилом — будет. Разрешимо ли образование подтипов в системе с таким правилом?

28.5. Неразрешимость полной $F_{<:}$

В предыдущем разделе мы установили, что алгоритмические правила образования подтипов для полной $F_{<:}$ корректны и полны — т. е., наименьшее отношение, замкнутое относительно этих правил, содержит те же самые утверждения, что и наименьшее отношение, замкнутое относительно исходных декларативных правил. Остается нерешенным вопрос о том, завершается ли алгоритм, реализующий эти правила, при всех возможных входах. К сожалению, это не так. Когда этот факт был обнаружен, многие были удивлены.

Упражнение 28.5.1 ($\star$)**.** Если алгоритмические правила для полной $F_{<:}$ не определяют алгоритм, который всегда завершается, то, очевидно, доказательство завершения для ядерной $F_{<:}$ невозможно перенести на правила полной системы. Где именно оно ломается?

Вот пример, найденный Гелли (Ghelli, 1995), который приводит к зацикливанию алгоритма проверки образования подтипов. Сначала определим следующее сокращение:

$$\neg S \forall X{<:}S.X.$$

Ключевое свойство оператора $\neg$ состоит в том, что он позволяет обменивать местами стороны утверждений об образовании подтипов.

Утверждение 28.5.2. $\Gamma \vdash \neg S <: \neg T$ тогда и только тогда, когда $\Gamma \vdash S <: T$.

Доказательство. Упражнение [$\star\star \nrightarrow$]. □

Определим теперь тип T следующим образом:

$$T = \forall X{<:}\mathsf{Top}.\ \neg(\forall Y{<:}X.\neg Y)$$

Если с помощью алгоритмических правил образования подтипов мы попробуем снизу вверх построить дерево вывода для утверждения

$$X_0{<:}T \quad \mapsto \quad X_0 \quad <: \quad \forall X_1{<:}X_0.\neg X_1$$

то мы получим бесконечную последовательность всё возрастающих подцелей:

$$\begin{array}{llll}
X_0{<:}T & \mapsto & X_0 & <: \quad \forall X_1{<:}X_0.\neg X_1 \\
X_0{<:}T & \mapsto & \forall X_1{<:}\mathsf{Top}.\ \neg(\forall X_2{<:}X_1.\neg X_2) & <: \quad \forall X_1{<:}X_0.\neg X_1 \\
X_0{<:}T, X_1{<:}X_0 & \mapsto & \neg(\forall X_2{<:}X_1.\neg X_2) & <: \quad \neg X_1 \\
X_0{<:}T, X_1{<:}X_0 & \mapsto & X_1 & <: \quad \forall X_2{<:}X_1.\neg X_2 \\
X_0{<:}T, X_1{<:}X_0 & \mapsto & X_0 & <: \quad \forall X_2{<:}X_1.\neg X_2 \\
\text{и т. д.} & & &
\end{array}$$

Шаги переименования, требуемые для сохранения корректности контекста, здесь производятся без специального указания, и имена переменных выбираются так, чтобы лучше была видна схема бесконечного регресса. Основная хитрость состоит в «смене границ», которая происходит, например, между второй и третьей строкой, где граница для X_1 в левой части, которая в строке 2 была равна Top, в строке 3 становится равной X_0. Поскольку в строке 2 вся левая сторона сама по себе является верхней границей для X_0, такая смена границ приводит к циклическому процессу, в каждой итерации которого контекст содержит все более и более длинные цепочки переменных. (Мы предупреждаем читателя, что в этом примере не стоит искать *семантического* смысла; в частности, $\neg T$ представляет собой отрицание только синтаксически.)

Хуже того, не только один определенный алгоритм зацикливается на некоторых входных данных, но можно также показать (Pierce 1994), что *не существует* алгоритма, корректного и полного относительно полной $F_{<:}$ и завершающегося на всех входных данных. Доказательство этой теоремы слишком длинно для данной книги. Однако, чтобы ощутить её общую идею, мы приведем еще один пример.

Определение 28.5.3. *Положительные* и *отрицательные* вхождения в типе T определяются следующим образом. Сам T является положительным вхождением в T. Если $T_1 \to T_2$ является положительным (или, соответственно, отрицательным) вхождением, то T_1 является отрицательным (или, соотв., положительным) вхождением, а T_2 является положительным (или, соотв., отрицательным) вхождением. Если $\forall X<:T_1.T_2$ — положительное (или, соотв., отрицательное) вхождение, то T_1 — отрицательное (соотв. положительное), а T_2 — положительное (или, соотв., отрицательное) вхождение. Положительные и отрицательные вхождения в утверждение об образовании подтипов $\Gamma \vdash S <: T$ определяются так: тип S и границы типовых переменных в Γ отрицательны, а тип T положителен.

Термины «положительный» и «отрицательный» происходят из логики. Согласно широко известному *соотношению Карри—Говарда* (Curry—Howard correspondence) между пропозициями и типами (§9.4), тип $S \to T$ соответствует логическому утверждению $S \Rightarrow T$, которое, по определению логической импликации, эквивалентно $\neg S \vee T$. Подутверждение S, очевидно, находится здесь в «отрицательной» позиции — а именно, внутри нечетного количества отрицаний — тогда и только тогда, когда вся импликация расположена внутри четного количества отрицаний. Заметим, что положительное вхождение в T соответствует отрицательному вхождению в $\neg T$.

Утверждение 28.5.4. Если X входит в S только положительно, а в T — только отрицательно, то $X<:U \vdash S<:T$ тогда и только тогда, когда $\vdash [X \mapsto U]S <: [X \mapsto U]T$.

Доказательство. Упражнение [$\star\star \nrightarrow$]. □

Пусть теперь T будет следующим типом:

$$\begin{aligned} T \;=\; & \forall X_0<:\mathsf{Top}.\forall X_1<:\mathsf{Top}.\forall X_2<:\mathsf{Top}. \\ & \neg(\forall Y_0<:X_0.\forall Y_1<:X_1.\forall Y_2<:X_2.\neg X_0) \end{aligned}$$

Рассмотрим утверждение об образовании подтипов

$$\begin{array}{lll} \vdash & \mathsf{T} & <: \quad \forall X_0{<:}\mathsf{T}.\forall X_1{<:}\mathsf{P}.\forall X_2{<:}\mathsf{Q}. \\ & & \quad \neg(\forall Y_0{<:}\mathsf{Top}.\forall Y_1{<:}\mathsf{Top}.\forall Y_2{<:}\mathsf{Top}. \\ & & \quad\ \neg(\forall Z_0{<:}Y_0.\forall Z_1{<:}Y_2.\forall Z_2{<:}Y_1.\mathsf{U})) \end{array}$$

Это утверждение можно рассматривать как описание состояния некоторого простого компьютера. Переменные X_1 и X_2 являются «регистрами» этой машины. Текущим состоянием этих регистров служат типы P и Q. «Поток команд» машины содержится в третьей строке: первая команда закодирована в границах (Y_2 и Y_1 — обратите внимание на порядок) для переменных Z_1 и Z_2, а непроясненный тип U представляет оставшиеся команды в программе. Тип T, вложенные отрицания и границы переменных X_0 и Y_0 играют приблизительно ту же роль, что и в более простом предыдущем примере: они позволяют нам «повернуть рычаг» и вернуться к подцели, имеющей ту же форму, что и исходная цель. Один поворот рычага будет соответствовать одному такту нашей машины.

В этом примере команда в начале потока команд кодирует собой инструкцию «обменять содержимое регистров 1 и 2». Чтобы убедиться в этом, мы с помощью двух установленных нами ранее утверждений проводим следующее вычисление. (Переменные P и Q, содержимое регистров, выделены, чтобы за ними было проще следить.)

$$\begin{array}{lll} & \vdash \quad \mathsf{T} & \\ & <: \quad \forall X_0{<:}\mathsf{T}.\forall X_1{<:}\mathbf{P}.\forall X_2{<:}\mathbf{Q}. & \\ & \qquad \neg(\forall Y_0{<:}\mathsf{Top}.\forall Y_1{<:}\mathsf{Top}.\forall Y_2{<:}\mathsf{Top}. & \\ & \qquad\ \neg(\forall Z_0{<:}Y_0.\forall Z_1{<:}Y_2.\forall Z_2{<:}Y_1.\mathsf{U})) & \\ \text{тит тк}^* & \vdash \quad \neg(\forall Y_0{<:}\mathsf{T}.\forall Y_1{<:}\mathbf{P}.\forall Y_2{<:}\mathbf{Q}.\neg\mathsf{T}) & \\ & <: \quad \neg(\forall Y_0{<:}\mathsf{Top}.\forall Y_1{<:}\mathsf{Top}.\forall Y_2{<:}\mathsf{Top}. & \text{по утверждению 28.5.4} \\ & \qquad \neg(\forall Z_0{<:}Y_0.\forall Z_1{<:}Y_2.\forall Z_2{<:}Y_1.\mathsf{U})) & \\ \text{тит тк} & \vdash \quad (\forall Y_0{<:}\mathsf{Top}.\forall Y_1{<:}\mathsf{Top}.\forall Y_2{<:}\mathsf{Top}. & \\ & \qquad \neg(\forall Z_0{<:}Y_0.\forall Z_1{<:}Y_2.\forall Z_2{<:}Y_1.\mathsf{U})) & \\ & <: \quad (\forall Y_0{<:}\mathsf{T}.\forall Y_1{<:}\mathbf{P}.\forall Y_2{<:}\mathbf{Q}.\neg\mathsf{T}) & \text{по утверждению 28.5.2} \\ \text{тит тк} & \vdash \quad \neg(\forall Z_0{<:}\mathsf{T}.\forall Z_1{<:}\mathbf{Q}.\forall Z_2{<:}\mathbf{P}.\mathsf{U}) & \\ & <: \quad \neg\mathsf{T} & \text{по утверждению 28.5.4} \\ \text{тит тк} & \vdash \quad \mathsf{T} & \\ & <: \quad (\forall Z_0{<:}\mathsf{T}.\forall Z_1{<:}\mathbf{Q}.\forall Z_2{<:}\mathbf{P}.\mathsf{U}) & \text{по утверждению 28.5.2} \end{array}$$

Заметим, что в конце этого вывода не только поменялись местами значения P и Q, но в процессе работы команда, которая вызвала этот обмен, была «истрачена», так что в начале потока «подлежащих исполнению» команд оказался тип U. Если

*«Тогда и только тогда, когда…» — *Прим. перев.*

теперь в качестве значения U мы выберем тип, имеющий такой же вид, как только что выполненная нами команда,

$$\begin{aligned} U \;=\; & \neg(\forall Y_0{<:}\mathsf{Top}.\forall Y_1{<:}\mathsf{Top}.\forall Y_2{<:}\mathsf{Top}. \\ & \quad \neg(\forall Z_0{<:}Y_0.\forall Z_1{<:}Y_2.\forall Z_2{<:}Y_1.U')) \end{aligned}$$

то мы проведем еще один обмен и вернем регистры к их исходному состоянию, прежде чем выполнить U′. Или же мы можем выбрать другое значение U, вызывающее какое-либо другое поведение. Например, если

$$\begin{aligned} U \;=\; & \neg(\forall Y_0{<:}\mathsf{Top}.\forall Y_1{<:}\mathsf{Top}.\forall Y_2{<:}\mathsf{Top}. \\ & \quad \neg(\forall Z_0{<:}Y_0.\forall Z_1{<:}Y_1.\forall Z_2{<:}Y_2.Y_1)) \end{aligned}$$

то на следующем такте исполнения машины текущее значение регистра 1, т. е. Q, окажется в позиции U — в сущности, будет произведен «косвенный переход» через регистр 1 к потоку команд, который представлен как Q. Обобщив этот прием, можно закодировать условные конструкции и арифметику (операции последователя, предшественника и проверку на ноль).

Собрав все это вместе, мы получаем доказательство неразрешимости путем сведения двухрегистровых машин — простого варианта обыкновенных машин Тьюринга, в которых есть конечное устройство управления и два счетчика, каждый из которых содержит натуральное число, — к утверждениям об образовании подтипов.

Теорема 28.5.5 (Пирс, 1994)**.** Для каждой двухрегистровой машины M существует такое утверждение об образовании подтипов $S(M)$, что $S(M)$ выводимо в полной $F_{<:}$ тогда и только тогда, когда вычисление M завершается.

Таким образом, если бы мы могли решить, доказуемо ли произвольное утверждение об образовании подтипов, мы могли бы также решить, останавливается ли произвольная двухрегистровая машина. Поскольку проблема останова для двухрегистровых машин неразрешима (см. Hopcroft and Ullman (1979)), неразрешима и задача определения образования подтипов для полной $F_{<:}$.

Следует еще раз подчеркнуть, что неразрешимость отношения образовании подтипов не означает, что полуалгоритм для образования подтипов, разработанный в §28.4, некорректен либо неполон. Если утверждение $\Gamma \vdash S <: T$ доказуемо согласно декларативным правилам для образования подтипов, то алгоритм определенно завершится и вернет значение *истина*. Если $\Gamma \vdash S <: T$ *не выводится* из декларативных правил, то алгоритм либо не завершится, либо вернет значение *ложь*. Каждое данное утверждение об образовании подтипов может так или иначе оказаться невыводимым: либо оно порождает бесконечную последовательность подцелей (что означает отсутствие конечного вывода с данным заключением), либо сводится к очевидному противоречию вроде $\mathsf{Top} <: S \to T$. Алгоритм проверки образования подтипов может распознать один из этих случаев, но не другой.

Означает ли неразрешимость полной $F_{<:}$, что система практически бесполезна? Напротив, как правило, считают, что *сама по себе* неразрешимость $F_{<:}$ не является таким уж серьезным недостатком. Во-первых, было показано (Ghelli, 1995), что, чтобы заставить процедуру проверки образования подтипов зациклиться, нужно дать ей цель с тремя достаточно экзотическими свойствами, причем трудно представить, что программист случайно напишет хоть какую-то из них. Кроме того, существует немало популярных языков, для которых задача проверки типов в принципе либо чрезвычайно трудоемка — как в ML или Haskell (§22.7), — либо вообще неразрешима, как для C++ или λProlog (Felty, Gunter, Hannan, Miller, Nadathur, and Scedrov, 1988). На практике оказывается, что отсутствие объединений и пересечений, о котором упоминается в следующем разделе (см. упражнение 28.6.3) является намного более серьезным недостатком полной $F_{<:}$, чем неразрешимость.

Упражнение 28.5.6 ($\star\star\star\star$). (1) Определите вариант полной $F_{<:}$ без типа Top, но со связываниями типов вида $X<:T$ и вида X (т. е. как с ограниченной, так и с неограниченной квантификацией); этот вариант называется *полностью ограниченной квантификацией* (completely bounded quantification). (2) Покажите, что отношение образования подтипов для этой системы разрешимо. (3) Дает ли такое ограничение удовлетворительное решение для проблем, обсуждаемых в этом разделе? В частности, будет ли оно работать в языках с дополнительными конструкторами типов, такими как числа, записи, варианты и т. п.?

28.6. Объединения и пересечения

В §16.3 мы убедились, что в языках с образованием подтипов желательно существование т. н. *объединения* для всякой пары типов S и T — то есть, типа J, минимального среди всех общих надтипов S и T. В этом разделе мы покажем, что отношение образования подтипов для ядерной $F_{<:}$ действительно имеет объединение для любой пары типов S и T, а также пересечение для любых S и T, имеющих хотя бы один общий подтип, и дадим алгоритмы для их вычисления. (Напротив, оба эти свойства отсутствуют в полной $F_{<:}$; см. упражнение 28.6.3.)

Мы пользуемся записью $\Gamma \vdash S \vee T = J$, означающей «$J$ является объединением типов S и T в контексте Γ», а также $\Gamma \vdash S \wedge T = M$, означающей «$M$ является пересечением S и J в Γ». Оба алгоритма для вычисления этих отношений определяются на рис. 28.5. Обратите внимание, что некоторые варианты в этих определениях пересекаются; чтобы определения работали как детерминистские алгоритмы, мы объявляем, что всегда выбирается первый подходящий вариант.

Несложно убедиться, что $\vee$ и $\wedge$ являются всюду определенными функциями в том смысле, что $\vee$ всегда возвращает тип, а $\wedge$ всегда либо возвращает тип, либо терпит неудачу. Для этого достаточно заметить, что общий вес (см. определение 28.3.4) типов S и T относительно Γ при рекурсивных вызовах всегда уменьшается.

$\to \forall <: \mathsf{Top}$

$\Gamma \vdash S \vee T =$

$$\begin{cases} T & \text{если } \Gamma \vdash S <: T \\ S & \text{если } \Gamma \vdash T <: S \\ J & \text{если } S = X \\ & \quad X<:U \in \Gamma \\ & \quad \Gamma \vdash U \vee T = J \\ J & \text{если } T = X \\ & \quad X<:U \in \Gamma \\ & \quad \Gamma \vdash S \vee U = J \\ M_1 \to J_2 & \text{если } S = S_1 \to S_2 \\ & \quad T = T_1 \to T_2 \\ & \quad \Gamma \vdash S_1 \wedge T_1 = M_1 \\ & \quad \Gamma \vdash S_2 \vee T_2 = J_2 \\ \forall X<:U_1.J_2 & \text{если } S = \forall X<:U_1.S_2 \\ & \quad T = \forall X<:U_1.T_2 \\ & \quad \Gamma, X<:U_1 \vdash S_2 \vee T_2 = J_2 \\ \mathsf{Top} & \text{в остальных случаях} \end{cases}$$

$\Gamma \vdash S \wedge T =$

$$\begin{cases} S & \text{если } \Gamma \vdash S <: T \\ T & \text{если } \Gamma \vdash T <: S \\ J_1 \to M_2 & \text{если } S = S_1 \to S_2 \\ & \quad T = T_1 \to T_2 \\ & \quad \Gamma \vdash S_1 \vee T_1 = J_1 \\ & \quad \Gamma \vdash S_2 \wedge T_2 = M_2 \\ \forall X<:U_1.M_2 & \text{если } S = \forall X<:U_1.S_2 \\ & \quad T = \forall X<:U_1.T_2 \\ & \quad \Gamma, X<:U_1 \vdash S_2 \wedge T_2 = M_2 \\ \textit{неудача} & \text{в остальных случаях} \end{cases}$$

Рис. 28.5. Алгоритмы поиска объединений и пересечений для ядерной $F_{<:}$

Теперь следует доказать, что по этим алгоритмам действительно вычисляются объединения и пересечения. Доказательство разбито на две части: утверждение 28.6.1 показывает, что вычисленное объединение является верхней гранью S и T, а пересечение (когда оно есть) является нижней гранью. Затем утверждение 28.6.2 показывает, что вычисленное объединение меньше любой верхней грани S и T, а пересечение больше любой общей нижней грани (и существует всегда, когда у S и T имеется общая нижняя грань).

Утверждение 28.6.1.

1. Если $\Gamma \vdash S \vee T = J$, то $\Gamma \vdash S <: J$ и $\Gamma \vdash T <: J$.
2. Если $\Gamma \vdash S \wedge T = M$, то $\Gamma \vdash M <: S$ и $\Gamma \vdash M <: T$.

Доказательство. Прямолинейная индукция по размеру вывода $\Gamma \vdash S \vee T = J$ или $\Gamma \vdash S \wedge T = M$ (т. е. по количеству рекурсивных вызовов, требуемых для вычисления J или M). □

Утверждение 28.6.2.

1. Если $\Gamma \vdash S <: V$ и $\Gamma \vdash T <: V$, то $\Gamma \vdash S \vee T = J$ для некоторого J, причем $\Gamma \vdash J <: V$.

2. Если $\Gamma \vdash L <: S$ и $\Gamma \vdash L <: T$, то $\Gamma \vdash S \wedge T = M$ для некоторого M, причем $\Gamma \vdash L <: M$.

Доказательство. Проще всего доказать обе части утверждения одновременной индукцией по размерам *алгоритмических* выводов утверждений $\Gamma \mapsto S <: V$ и $\Gamma \mapsto T <: V$ для части 1, и $\Gamma \mapsto L <: S$ и $\Gamma \mapsto L <: T$ для части 2. (Благодаря теореме 28.3.3 мы можем быть уверены, что алгоритмические соответствия декларативным выводам всегда существуют.)

1. Если какой-либо из двух выводов является экземпляром правила SA-Top, то $V = \mathrm{Top}$, и требуемый результат, $\Gamma \vdash J <: V$, следует непосредственно.

Если вывод $\Gamma \mapsto T <: V$ является экземпляром правила SA-Refl-TVar, то $T = V$. Но в таком случае первый данный вывод дает нам $\Gamma \mapsto S <: V = T$, так что применим первый вариант в определении объединения, и он дает нам $\Gamma \vdash S \vee T = T$, что удовлетворяет требованиям. Аналогично, если вывод $\Gamma \mapsto S <: V$ является экземпляром SA-Refl-TVar, то $S = V$. Но тогда второй данный вывод говорит, что $\Gamma \vdash T <: V = S$, так что применим второй вариант в определении объединения, и он дает нам $\Gamma \vdash S \vee T = S$, что опять же удовлетворяет требованиям.

Если вывод $\Gamma \mapsto S <: V$ заканчивается экземпляром правила SA-Trans-TVar, то мы имеем $S = X$, причем $X<:U \in \Gamma$, и имеется подвывод $\Gamma \vdash U \vee T = J$. Третий вариант в определении объединения дает $\Gamma \vdash S \vee T = J$, а из индуктивного предположения мы имеем $\Gamma \vdash J <: V$. Аналогично мы рассуждаем и в том случае, когда $\Gamma \mapsto T <: V$ заканчивается на SA-Trans-TVar.

Теперь по форме алгоритмических правил образования подтипов несложно видеть, что остаются только варианты, в которых оба данных вывода завершаются либо правилом SA-Arrow, либо SA-All.

Если оба вывода завершаются SA-Arrow, то мы имеем $S = S_1 \to S_2$, $T = T_1 \to T_2$ и $V = V_1 \to V_2$, причем $\Gamma \mapsto V_1 <: S_1$, $\Gamma \mapsto S_2 <: V_2$, $\Gamma \mapsto V_1 <: T_1$ и $\Gamma \mapsto T_2 <: V_2$. Согласно части (2) предположения индукции, $\Gamma \vdash S_1 \wedge T_1 = M_1$ для некоторого M_1, причем $\Gamma \vdash V_1 <: M_1$, а согласно части (1), $\Gamma \vdash S_2 \vee T_2 = J_2$ для некоторого J_2, причем $\Gamma \vdash J_2 <: M_2$. Пятый вариант в определении объединений дает $\Gamma \vdash S_1 \to S_2 \vee T_1 \to T_2 = M_1 \to J_2$, а по правилу S-Arrow мы имеем $\Gamma \vdash M_1 \to J_2 <: V_1 \to V_2$.

Наконец, если оба данных вывода заканчиваются на SA-All, то мы имеем $S = \forall X<:U_1.\, S_2$, $T = \forall X<:U_1 \to T_2$ и $V = \forall X<:U_1.\, V_2$, причем $\Gamma, X <: U_1 \mapsto S_2 <: V_2$ и $\Gamma, X <: U_1 \mapsto T_2 <: V_2$. Согласно части (1) предположения индукции, $\Gamma, X<:U_1 \vdash S_2 \vee T_2 = J_2$, причем $\Gamma, X<:U_1 \mapsto J_2 <: V_2$. Шестой вариант в определении объединений дает нам $J = \forall X<U_1.\, J_2$, а по правилу S-All имеем $\Gamma \vdash \forall X<:U_1.\, J_2 <: \forall X<:U_1.\, V_2$.

2. Если вывод $\Gamma \mapsto L <: T$ заканчивается на SA-Top, то тип T равен Top, так что $\Gamma \vdash S <: T$, и, по первому варианту в определении пересечения, $\Gamma \vdash S \wedge T = S$. Однако из второго данного вывода мы знаем, что $\Gamma \vdash L <: S$, так что требуемое утверждение доказано. Аналогично рассуждаем и в случае, когда вывод $\Gamma \mapsto L <: S$ заканчивается на SA-Top.

Если вывод $\Gamma \mapsto L <: S$ заканчивается на SA-Refl-TVar, то $L = S$, и второй данный вывод дает $\Gamma \vdash L = S <: T$, откуда по определению пересечения мы имеем $\Gamma \vdash S \wedge T = S$, так что доказательство закончено. Аналогично рассуждаем, когда вывод $\Gamma \mapsto L <: T$ заканчивается на SA-Refl-TVar.

Остаются только варианты, в которых оба данных вывода заканчиваются на SA-Trans-TVar, SA-Arrow или SA-All.

Если оба вывода заканчиваются на SA-Trans-TVar, то имеем $L = X$, причем $X<:U \in \Gamma$, и имеются два подвывода $\Gamma \mapsto U <: S$ и $\Gamma \mapsto U <: T$. Согласно части (2) предположения индукции, $\Gamma \vdash U <: M$, откуда имеем $\Gamma \vdash L <: M$ по правилу S-TVar и транзитивности.

Если оба вывода заканчиваются на SA-Arrow, то мы имеем $S = S_1 \to S_2$, $T = T_1 \to T_2$ и $L = L_1 \to L_2$, причем $\Gamma \mapsto S_1 <: L_1$, $\Gamma \mapsto L_2 <: S_2$, $\Gamma \mapsto T_1 <: L_1$ и $\Gamma \mapsto L_2 <: T_2$. Согласно части (1) предположения индукции, $\Gamma \vdash S_1 \vee T_1 = J_1$ для некоторого J_1, причем $\Gamma \vdash J_1 <: L_1$, а согласно части (2), $\Gamma \vdash S_2 \wedge T_2 = M_2$, причем $\Gamma \vdash L_2 <: M_2$. Определение пересечений дает нам $\Gamma \vdash S_1 \to S_2 \wedge T_1 \to T_2 = J_1 \to M_2$, а по правилу S-Arrow, $\Gamma \vdash L_1 \to L_2 <: J_1 \to M_2$.

Аналогично ведется доказательство и в варианте с правилом SA-All. □

Упражнение 28.6.3 (Рекомендуется, ⋆⋆⋆). Рассмотрим пару типов, предложенных Гелли (Ghelli, 1990): $S = \forall X<:Y \to Z.Y \to Z$ и $T = \forall X<:Y' \to Z'.Y' \to Z'$ и контекст $\Gamma = Y<:\mathsf{Top}, Z<:\mathsf{Top}, Y'<:Y, Z'<:Z$. (1) Сколько в полной $F_{<:}$ существует типов, являющихся подтипами как S, так и T в контексте Γ? (2) Покажите, что в полной $F_{<:}$ типы S и T не имеют пересечения в контексте Γ. (3) Найдите пару типов, которые в полной $F_{<:}$ не имеют объединения в контексте Γ.

28.7. Ограниченные кванторы существования

Чтобы расширить алгоритм ядерной $F_{<:}$ на язык с кванторами существования, нужно преодолеть одну дополнительную трудность. Напомним декларативное правило устранения для экзистенциальных типов:

$$\frac{\Gamma \vdash t_1 : \{\exists X<:T_{11}, T_{12}\} \qquad \Gamma, X<:T_{11}, x:T_{12} \vdash t_2 : T_2}{\Gamma \vdash \mathsf{let}\ \{X,x\}=t_1\ \mathsf{in}\ t_2 : T_2} \qquad \text{(T-Unpack)}$$

В §24.1 мы отметили, что типовая переменная X присутствует в контексте, в котором во второй предпосылке вычисляется тип терма t_2, но *отсутствует* в контексте заключения правила. Это означает, что тип T_2 не должен содержать X в качестве свободной переменной, поскольку всякое такое вхождение в заключении окажется вне области видимости. Более подробно этот вопрос обсуждался в §25.5, где мы указали, что изменение контекста от предпосылки к заключению соответствует *отрицательному* сдвигу индексов переменных в T_2, если мы представим термы в безымянном формате де Брауна; этот сдвиг будет неудачен, если X содержится в T_2 как свободная переменная.

Как это обстоятельство влияет на алгоритм минимальной типизации для языка с экзистенциальными типами? В частности, что нам делать с таким выражением, как $t = \mathsf{let}\ \{X,x\} = p\ \mathsf{in}\ x$, в котором p имеет тип $\{\exists X, \mathsf{Nat} \to X\}$? Наиболее естественным типом тела x будет $\mathsf{Nat} \to X$, упоминающий связанную переменную X. Однако согласно декларативному отношению типизации (с правилом включения), x имеет также типы $\mathsf{Nat} \to \mathsf{Top}$ и Top. Поскольку ни в одном из этих типов X не встречается,

всему терму t можно в декларативной системе присвоить типы Nat→Top и Top. В общем случае, в выражении распаковки всегда можно расширить тело до типа, в котором не содержится свободной переменной X, а затем применить T-Unpack. Так что, если мы хотим, чтобы наш алгоритм минимальной типизации был полон, он должен не просто объявлять об ошибке при работе с выражением распаковки, где минимальный тип тела T_2 содержит свободное вхождение связанной переменной X. Вместо этого он должен попытаться расширить T_2 до некоторого типа, в котором X не упоминается. Ключевое наблюдение, необходимое для того, чтобы такая идея сработала, состоит в том, что множество надтипов данного типа, не содержащих X, всегда имеет минимальный элемент. Это показывает следующее упражнение (решение которого было найдено Гелли и Пирсом, Ghelli and Pierce, 1998).

Упражнение 28.7.1 (⋆⋆⋆)**.** Постройте алгоритм, вычисляющий в ядерной $F_{<:}$ с ограниченными экзистенциальными типами минимальный надтип данного типа T, не содержащий переменной X, по отношению к данному контексту Γ. Такой надтип обозначается $R_{X,\Gamma}(T)$.

Теперь алгоритмическое правило типизации для устранения кванторов существования можно записать так:

$$\frac{\Gamma \vdash t_1 : T_1 \qquad \Gamma \vdash T_1 \Uparrow \{\exists X{<:}T_{11}, T_{12}\} \qquad \Gamma, X{<:}T_{11}, x{:}T_{12} \vdash t_2 : T_2 \qquad R_{X,(\Gamma,X<:T_{11},x:T_{12})}(T_2) = T_2'}{\Gamma \vdash \text{let } \{X,x\}{=}t_1 \text{ in } t_2 : T_2'} \tag{TA-Unpack}$$

Как и следовало ожидать, в *полной* $F_{<:}$ с ограниченными кванторами существования ситуация сложнее. Гелли и Пирс (Ghelli and Pierce, 1998) приводят пример типа T, контекста Γ и переменной X, таких, что множество надтипов T, не содержащих X, не имеет минимального элемента в контексте Γ. Отсюда немедленно следует, что отношение типизации в такой системе не обладает минимальными типами.

Упражнение 28.7.2 (⋆⋆⋆)**.** Покажите, что отношение образования подтипов для варианта полной $F_{<:}$, имеющего только кванторы существования (т. е. без кванторов общности) также неразрешимо.

28.8. Ограниченная квантификация и тип Bot

Добавление минимального типа Bot несколько усложняет метатеоретические свойства $F_{<:}$. Это происходит оттого, что в типе вида ∀X<:Bot.T внутри T переменная X является, на самом деле, *синонимом* Bot, поскольку по предположению X — подтип Bot, а Bot, по правилу S-Bot, — подтип X. Это, в свою очередь, означает, что пары типов вроде ∀X<:Bot.X→X и ∀X<:Bot.Bot→Bot эквивалентны в отношении образования подтипов, хотя синтаксически они различаются. Более того, если окружающий контекст содержит предположения X<:Bot и Y<:Bot, то типы X→Y и Y→X

эквивалентны, хотя ни в одном из них Bot явно не упоминается. Несмотря на все это, даже при наличии Bot все основные свойства $F_{<:}$ все равно сохраняются. Детали можно найти в Pierce (1997a).