Линейная темпоральная логика
Проверка свойств на моделях (model checking)

Модальные логики

- Модальность (лат. modus — способ, вид) — способ, вид бытия или события; категории модальности: возможность, действительность, необходимость.
- Расширяем язык: как бы выразить «модальности»? *Всегда* зимой идёт снег. Дождь *может* идти при солнечном свете.
- Модифицируем язык, модифицируем аксиоматику, модифицируем теорию моделей.
- Язык предполагает включение новых связок, самые типичные:

  $\Box$ необходимость (necessity)

  $\Diamond$ возможность (possibility)
- Интуитивный смысл связок примерно понятен, конкретный смысл формализуется в конкретной теории.

Некоторые модальные исчисления (обзор)

Терминология введена Кларенсом Льюисом и Купером Лангфордом в 1932 году.

- Минимальная модальная логика (K) строится поверх ИВ:

Аксиомы ИВ $\quad \Box(\varphi \to \psi) \to (\Box\varphi \to \Box\psi) \quad \dfrac{\varphi \quad \varphi \to \psi}{\psi} \quad \dfrac{\varphi}{\Box\varphi}$

- K4: K и дополнительная аксиома транзитивности $\Box\varphi \to \Box\Box\varphi$;
- T: K и дополнительная аксиома рефлексивности $\Box\varphi \to \varphi$;
- S4: K4 + T;
- S5: T и аксиома $\Diamond\varphi \to \Box\Diamond\varphi$ («если что-то возможно, то оно обязательно реализуется в каком-то мире»). Например, утверждение «если Бог возможен, то он необходимо существует» можно формализовать и доказать в S5.

Теория моделей

- Много разных — например, топологические интерпретации (как и в ИИВ).
- Наш интерес сегодня — интерпретации на многих мирах (Саул Крипке), где миры упорядочены некоторым отношением. Тогда интуитивный смысл связок: $\Box$ — истинно во всех достижимых мирах, $\Diamond$ — истинно в каком-то достижимом мире. Можем указывать конкретный мир W:

$$W \models \Box A$$

- Скажем, следующее выполнено в интуиционистских моделях:

$$W \models X \to \Box X \qquad W \models (\alpha \to \beta) \to (\Box(\alpha \to \beta)) \qquad W \models \neg\alpha \to \neg\Diamond\alpha$$

Линейная темпоральная логика

- Темпоральная логика: множественные миры (в стиле моделей Крипке) понимаются как расположенные в соответствие с течением времени.
- Линейная темпоральная логика: миры выстроены в линейном порядке.
- Используем следующие связки:
  1. $\mathcal{G}(\alpha)$ или $\Box\alpha$: утверждение α выполнено в любой момент (начиная с текущего).
  2. $\mathcal{P}(\alpha)$ или $\bigcirc\alpha$: утверждение α выполнено в следующий момент.
  3. $\mathcal{E}(\alpha)$ или $\Diamond\alpha$: утверждение α неизбежно выполнено в будущем, в какой-то момент (начиная с текущего).
  4. $\mathcal{U}(\alpha, \beta)$ или $\alpha\mathcal{U}\beta$: утверждение α истинно, пока β не станет истинным, после чего α может быть любым.

Представление моделей ЛТЛ как множества слов

$$W(\varphi) = \{\sigma \in (\mathcal{P}(a))^\omega \mid \sigma \models \varphi\}$$

На строке $\sigma = S_0 S_1 S_2 \ldots$ (каждый $S_i \subseteq \mathcal{P}(a)$) истинность задаётся так:

$\sigma \models \top$	всегда
$\sigma \models a$	$a \in S_0$
$\sigma \models \varphi_1 \mathbin{\&} \varphi_2$	$\sigma \models \varphi_1$ и $\sigma \models \varphi_2$
$\sigma \models \neg\varphi$	$\sigma \not\models \varphi$
$\sigma \models \bigcirc\varphi$	$\sigma[1\ldots] \models \varphi$
$\sigma \models \varphi_1 \mathcal{U} \varphi_2$	существует $j.\sigma[j\ldots] \models \varphi_2$ и при всех $i, 0 \leqslant i < j.\sigma[i\ldots] \models \varphi_1$

Выразимость связок, другие формулы

Будем рассматривать следующую грамматику для формул:

$$\varphi ::= \top \mid a \mid \varphi \,\&\, \varphi \mid \neg\varphi \mid \bigcirc \varphi \mid \varphi \mathcal{U} \varphi$$

поскольку остальные связки выражаются через эти.
В самом деле, имеем следующие тождества:

- Связки выражаются друг через друга:
  - $\Box\alpha = \neg\Diamond\neg\alpha$
  - $\Diamond\alpha = \top\mathcal{U}\alpha$
- Правила двойственности:
  - $\neg \bigcirc \varphi = \bigcirc\neg\varphi$
  - $\neg\Diamond\varphi = \Box\neg\varphi$
  - $\neg\Box\varphi = \Diamond\neg\varphi$
- Правила расширения:
  - $\varphi\mathcal{U}\psi = \psi \vee (\varphi \,\&\, \bigcirc(\varphi\mathcal{U}\psi))$
  - $\Diamond\varphi = \varphi \vee \bigcirc\Diamond\varphi$
  - $\Box\varphi = \varphi \,\&\, \bigcirc\Box\varphi$

Задача проверки программы на моделях

- Проверить многопоточный алгоритм/протокол — как?
- Первоначальная идея: Clarke E., Emerson E., Sistla A., Automatic verification of finite-state concurrent systems using temporal logic specifications 1986.
- Опишем состояние некоторым набором (булевских) переменных. Программа задаёт набор допустимых переходов между состояниями.
- Запишем условие (которое желаем проверить) как формулу ЛТЛ.
- Научимся проверять выполнимость формулы на данном множестве состояний.

Постановка задачи

Определение

Системой переходов назовём граф состояний, в котором каждое состояние отражает содержимое памяти компьютера, а переходы соответствуют инструкциям (операциям), выполняемым компьютером

Хотим научиться проверять, выполнено ли φ при всех возможных вариантах выполнения программы, то есть при всех возможных путях в системе переходов *TS*:

$$TS \models \varphi$$

Очевидно, *TS* задаёт некоторое множество (бесконечных) строк в алфавите $2^{FV(\varphi)}$. Находится ли в этом множестве строка, удовлетворяющая φ?

Недетерменированные (обобщённые) автоматы Бюхи

Определение

НАБ (НОАБ) определяется внешним алфавитом A, множествами состояний Q, функцией переходов $\delta : A \times Q \to \mathcal{P}(Q)$ и семейством допускающих множеств состояний $\mathcal{F} \subseteq \mathcal{P}(Q)$.
Бесконечная строка $\alpha = a_0 a_1 a_2 \dots$ допускается недетерменированным (обобщённым) автоматом Бюхи, если найдётся такая последовательность состояний $q_0 q_1 q_2 \dots$, что $q_{n+1} \in \delta(a_n, q_n)$ и в процессе применения автомата к ней каждое из множеств допускающих состояний будет посещено бесконечное количество раз.

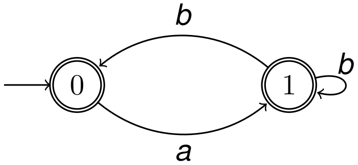
$$\forall F \in \mathcal{F}. \forall n \in \mathbb{N}. \exists m > n. q_m \in F$$

В случае $|\mathcal{F}| \leqslant 1$ такой автомат — НАБ, иначе — НОАБ.

Пример обобщённого недетерменированного автомата Бюхи

Пример

Рассмотрим автомат с двумя состояниями (0,1), начальным состоянием 0, и $\mathcal{F} = \{\{0\}, \{1\}\}$.



Строка $(ab)^\omega$ *будет принята, строка* $a(b^\omega)$ *— нет.*

Строим автомат Бюхи для формулы: состояния

- Раскроем сокращения записи (выразим $\Box$, $\vee$, $\rightarrow$ и $\Diamond$ через другие связки).
- Рассмотрим $\mathcal{B}(\varphi)$ — семейство всех подформул φ (и их отрицаний, с учётом $\varphi = \neg\neg\varphi$), образующих непротиворечивое максимальное множество. Скажем, для $a \vee \neg b$ это будет $\{\{\neg a, \neg b, a \vee \neg b\}, \{a, \neg b, a \vee \neg b\}, \{\neg a, b, \neg(a \vee \neg b)\}, \{a, b, a \vee \neg b\}\}$
- Поскольку множество содержит модальные операторы, непротиворечивость также должна соответствовать условиям реализуемости: при рассмотрении подформулы $\varphi\mathcal{U}\psi$ должно быть

$$\psi \in B \Rightarrow \varphi\mathcal{U}\psi \in B \qquad \varphi\mathcal{U}\psi \in B \text{ и } \psi \notin B \Rightarrow \varphi \in B$$

- Состояния автомата — $B_n \in \mathcal{B}(\varphi)$.
- Стрелки подписаны состоянием переменных A, и их может быть несколько одинаково подписанных, поскольку в силу модального характера значение формулы не исчерпывается значением переменных.

Строим автомат Бюхи для формулы: переходы

Рассмотрим состояние B и набор переменных $A \in \mathcal{P}(FV(\varphi))$.

- Рассмотрим $B : A = B \cup FV(\varphi)$ — состояния, в которых пропозициональные переменные соответствуют ожидаемому набору переменных.
- Тогда $B \in \delta(A, B)$, если и только если для каждого из $\psi \in B$ выполнено одно из следующих условий:
  1. если ψ — не модальный оператор (здесь неявная рекурсия по структуре ψ), то $\psi \in B'$;
  2. если $\psi \equiv \bigcirc\varphi$, то $\varphi \in B'$;
  3. если $\psi \equiv \varphi_1 \mathcal{U} \varphi_2$, то выполнен закон расширения для $\mathcal{U}$:

     либо φ_2 B ($\mathcal{U}$ активирован в текущем состоянии).
     либо φ_1 B и $\varphi_1 \mathcal{U} \varphi_2$ B ($\mathcal{U}$ будет активирован позже).

Строим автомат Бюхи для формулы: допускающие состояния

Автомат для формулы φ:

$$\begin{aligned}
Q \quad &:= \mathcal{B}(\varphi) \\
Q_0 \quad &:= \{B \mid \varphi \in B, B \in Q\} \\
F_{\psi_1 \mathcal{U} \psi_2} \quad &:= \{B \mid \psi_1 \mathcal{U} \psi_2 \notin B \text{ или } \psi_2 \in B\}, \\
\mathcal{F} \quad &:= \{F_{\psi_1 \mathcal{U} \psi_2}\}
\end{aligned}$$

Идея в том, что автомат окажется в допускающем состоянии относительно $\psi_1 \mathcal{U} \psi_2$:

- либо, когда формула $\psi_1 \mathcal{U} \psi_2$ не нужна для результата;
- либо в тот момент, когда соответствующая формула $\psi_1 \mathcal{U} \psi_2$ «активируется» — оператор меняет фокус восприятия с ранее истинного ψ_1 на истинный ψ_2.

Почему автомат Бюхи?

Напомним, обобщённый автомат Бюхи принимает строку, если все допускающие множества состояния посещаются при обработке строки бесконечное количество раз. Соответственно:

- если автомат имеет $\mathcal{F} = \varnothing$, то он примет любую бесконечную последовательность переходов.
- если автомат построен для оператора $\mathcal{U}$, и оператор активируется на шаге *k* в последний раз — значит, на шаге $k + 1$ и на последующих шагах данное выражение не будет истинным (для истинности $\varphi\mathcal{U}\psi$ требуется наличия момента активации в будущем).

Разрешимость задачи TS φ в ЛТЛ

Идея алгоритма.

1. Построим обобщённый недетерминированный автомат Бюхи для формулы $\neg\varphi$ (принимает последовательность исполнения тогда и только тогда, когда она опровергает φ);
2. построим недетерминированный автомат по системе переходов TS;
3. построим их пересечение — и преобразуем автомат в недетерменированный автомат Бюхи T (с множеством допускающих состояний F).
4. проверим $\mathcal{L}_T = \varnothing$ (значит, φ доказано), либо найдём контрпример — последовательность исполнения, имеющая цикл, затрагивающий состояние из F (обход графа состояний; алгоритм заканчивается в силу конечности множества состояний);
5. данная последовательность будет контрпримером к задаче $TS \models \varphi$.

$\square$

Пример реализации: SPIN

- Один из первых инструментов (1991 год).
- Используется специальный язык для описания алгоритмов/протоколов (Promela).
- Язык позволяет формализовать параллельные вычисления.
- Программа может быть вычислена в разных окружениях (например, случайное исполнение).
- Также, к программе могут быть добавлены условия, которые либо будут доказаны — либо будет найден контрпример (контрпример будет предъявлен).

Пример программы на Promela: каковы возможные значения n?

```
byte n = 0, finish = 0;
active [2] proctype P() {
    byte register, counter = 0;
    do :: counter = 10 -> break
       :: else ->
              register = n;
              register++;
              n = register;
              counter++
    od;
    finish++
}
active proctype WaitForFinish() {
    finish == 2;
    printf(“n = %d\n”, n)
}
```