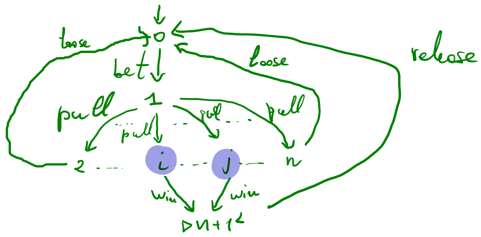
Example. A (simplified) 3-wheels slot machine



$S = \{0, \dots, n+1\}$ & $I = \{0\}$

$Act = \{bet, win, loose, pull, release\}$

Fix an interval $[i,j]$ with $2 \leq i \leq j \leq n$

$$\rightarrow \; = \{(0, bet, 1)\} \cup \bigcup_{h \in [i,j]} (\{(1, pull, h), (h, win, n+1)\}) \cup \bigcup_{h \in S \setminus [i,j]} \{(h, loose, 0)\} \cup \{(n+1, release, 0)\}$$

3 wheels

$$AP = \bigcup_{f \in Fruits} \{w_1 = f, w_2 = f, w_3 = f\} \cup \{price = p \mid p \in [i,j]\}$$ where $Fruits = \{apple, pear, banana \dots\}$

let $c : [i,j] \longrightarrow Fruit^3$

$$L : h \longmapsto \{price = h, w_1 = f_1, w_2 = f_2, w_3 = f_3 \mid c(h) = (f_1, f_2, f_3)\}$$

Exercise: Define L on $h \notin \{i, \dots, j\}$

Non-determinism

- crucial modelling mechanism (e.g., pull transitions from 0 in the slot machine)
- under-specification

Deterministic TS $|I| \leq 1$

- **action-deterministic** $\forall s \in S, \alpha \in Act : |Post(s, \alpha)| \leq 1$

  $Post(s,\alpha) := \{s' \in S : s \xrightarrow{\alpha} s'\}$

  $s \xrightarrow{\alpha} s_1$, $s \xrightarrow{\alpha} s_2$, $s_1 = s_2$

- **AP-deterministic** $\forall s \in S \; \forall A \in 2^{AP} : |\{s' \in Post(s) \mid L(s') = A\}| \leq 1$

  $Post(s) := \bigcup_{\alpha \in Act} Post(s, \alpha)$

  $s \xrightarrow{\alpha} s_1$, $s \xrightarrow{\beta} s_2$, $L(s_1) = L(s_2) \Rightarrow s_1 = s_2$

Executions / Traces

Execution fragment $\rho \in \underbrace{S(Act\, S)^*}_{\text{finite}} \cup \underbrace{S(Act\, S)^\omega}_{\text{infinite}}$

s.t. $\rho = s_0 \alpha_1 s_1 \alpha_2 s_2 \dots \alpha_n s_n \dots \Rightarrow$ for all i: $s_i \xrightarrow{\alpha_{i+1}} s_{i+1}$

ρ **maximal** if ρ infinite or
$\rho = s_0 \alpha_1 s_1 \alpha_2 s_2 \dots \alpha_n s_n \wedge Post(s_n) = \emptyset$

ρ **initial** if $s_0 \in I$

Execution initial & maximal execution fragment.

Reachable states $Reach(TS) = \{s \mid \exists \rho$ initial execution fragment ending in $s\}$

A note inspired by Duncan Attard's question (ay 20/21)
"Why do we need both labelling & actions to express properties?":

Verification can be

- action-based
- state-based
- action+state based — this is more involved

Execution (fragments) are used for action-based verification; this is the usual approach when it is necessary to model interactions.

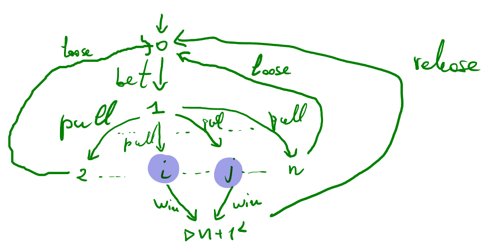
We are now going to see a state-based approach, where algorithms "ignore" actions. Formally:

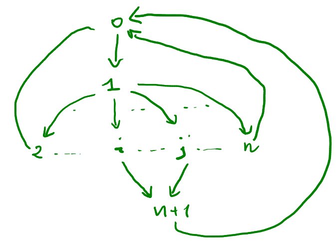
the state graph of $TS = (S, Act, \rightarrow, I, AP, L)$ is obtained by "removing" the actions from TS

$$G(TS) = \langle S, E \rangle \quad \text{where} \quad E = \bigcup_{s \in S} \{s\} \times post(s)$$

Example

the TS of the slot machine & its state graph





note that state label also "disappear" but that's a sort of illusion :)

> Notation: given a sequence $\sigma = \sigma_0 \sigma_1 \dots \sigma_n \dots$
> - $|\sigma|$ is its lenght (if σ is infinite, $|\sigma| = \infty$)
> - $\sigma[i]$ is the i-th element of σ
> - if σ is finite then $last(\sigma)$ is the last element of σ

From now on we assume TS fixed.

Linear Time Beh & Properties

A **path fragment of TS** is a path in its state graph:

$\pi \in S^* \cup S^\omega$ s.t. $\forall\, 0 \le i < |\pi| : \pi[i+1] \in Post(\pi[i])$

π maximal — if $\pi \in S^*$ & $Post(last(\pi)) = \emptyset$ or $\pi \in S^\omega$

π initial — if $\pi[0] \in I$

π path — if initial & maximal

$\bigcup_{\{\pi \text{ path}(TS) : \pi[0] = s\}} trace(\pi)$

Trace of π $\{L(\pi[i])\}_{0 \le i < |\pi|}$

$$\boxed{Traces(TS) := \bigcup_{s \in I} traces(s)}$$

An **LT property** (on AP) is an element P of $2^{(2^{AP})^\omega}$ i.e. $P \subseteq (2^{AP})^\omega$

Examples. Let $AP = \{red, green, yellow\}$ and P_{light} = "the traffic light is infinitely often red"

$P_{light} \ni \{red\}\{red, yellow\}\{green, yellow\}\{red\}\{red, yellow\}\{green, yellow\}$

$\not\ni \{red\}\{green\}\emptyset\,\emptyset\,\emptyset \ldots$

$\ni \{\{red\}\}^\omega$

$\ni X^\omega$ if $red \in X \subseteq AP$

$\ni \{X_i\}_{i \in \omega}$ if $red \in X_i \iff i$ prime

Let $AP = \{c_1, \ldots, c_n\}$ — thread h is in the critical section

$P_{mutex} = \{\{A_i\}_{i \in \omega} \in (2^{AP})^\omega \mid \forall i \ge 0,\ 1 \le h < k \le n : \{c_h, c_k\} \subseteq A_i \Rightarrow h = k\}$

$= \bigwedge_{1 \le h < k \le n} \{c_h, c_k\} \not\subseteq A_1 \wedge \ldots \wedge \bigwedge_{1 \le h < k \le n} \{c_h, c_k\} \not\subseteq A_n$

Exercise: What does $P' = \{\{A_i\}_{i \ge 0} \in (2^{AP})^\omega \mid \forall i \ge 0\ \exists\, 1 \le h \le n.\ c_h \in A_i\}$ state?
Give two different traces in P'

Exercise: Let P_{slot} : "always(price = 0 $\to$ eventually $\bigvee_{p \in [i,j]}$ price = p)". Give an example of an element of P_{slot} and one of $(2^{AP})^\omega \setminus P_{slot}$

$$
\begin{array}{cccccccc}
TS & I & & & s_i \xrightarrow{\alpha_i} s_{i+1} & & & \\
\cup & \cup & & & & & & \\
\pi = & s_0 & \alpha_1 & s_2 & \cdots\ \alpha_n & s_n & \alpha_{n+1} & \cdots \\
 & \downarrow & & \downarrow & & \downarrow & & \\
 & L(s_0) & & L(s_2) & \cdots & L(s_n) & \cdots & \in P \text{ LT property}
\end{array}
$$

$TS \models P$

The importance of Traces

WLOG: no terminal states in TS (hence all maximal paths are infinite)

the trace of a maximal path of TS is $\text{trace}(\pi) = \{L(\pi[i])\}_{i \geq 0}$

Notice that $\text{trace}(\pi) \in (2^{AP})^\omega$

$$\boxed{TS \models P \iff \text{Traces}(TS) \subseteq P}$$

$s \in S$, $s \models P \iff \text{traces}(s) \subseteq P$

Read $\text{Traces}(TS) \subseteq \text{Traces}(TS')$ as "TS correctly implements TS'"

refinement ($\text{Traces}(TS)$) abstract model ($\text{Traces}(TS')$)

Thm TS & TS' t.s. on the same atomic propositions then
$\text{Traces}(TS) \subseteq \text{Traces}(TS') \iff \forall$ LT prop. P $TS' \models P \Rightarrow TS \models P$

Proof ($\Rightarrow$) $TS' \models P \overset{\text{def}}{\iff} \text{Traces}(TS') \subseteq P$
$\overset{\text{hyp}}{\Longrightarrow} \text{Traces}(TS) \subseteq P \overset{\text{def}}{\iff} TS \models P$

($\Leftarrow$) $TS' \models \text{Traces}(TS')$ since $\text{Traces}(TS') \subseteq \text{Traces}(TS')$
$\overset{\text{hyp}}{\Longrightarrow} TS \models \text{Traces}(TS') \overset{\text{def}}{\iff} \text{Traces}(TS) \subseteq \text{Traces}(TS')$ □

Cor $\text{Traces}(TS) = \text{Traces}(TS') \iff \forall P$ LT f.la : $TS \models P \iff TS' \models P$

Exercise: Is