Tomasz Maciejewski

Semantyka gier

notatki

1. Czym jest gra?

Semantyka gier jest modelowaniem zachowania programu za pomocą gier. Gra jest pewną interakcją między graczem a adwersarzem. W semantyce języków programowania graczem jest program, a adwersarzem — środowisko.

Definicja 1. *Gra G to trójka (M_G, λ_G, P_G), gdzie:*
- *M_G to zbiór wszystkich możliwych ruchów,*
- *λ_G to funkcja typu $M_G \to \{O, P\}$,*
- *$P_G \subseteq M_G^{\star}$ taki, że:*

$$P_G \neq \emptyset, \tag{1}$$

$$\begin{aligned}\forall s \in P_G, i \in \mathbb{N}, 1 \leqslant i \leqslant |s| \,.\, (\mathit{even}(i) \Rightarrow \lambda_G(s_i) = P) \\ \wedge (\mathit{odd}(i) \Rightarrow \lambda_G(s_i) = O),\end{aligned} \tag{2}$$

$$\forall p_1, p_2, s \,.\, s \in P_G \wedge p_1 p_2 = s \Rightarrow p_1 \in P_G. \tag{3}$$

Własność 2 mówi o tym, że ruchy mają być wykonywane naprzemienne przez gracza i adwersarza, przy czym ten drugi zawsze ma zaczynać rozgrywkę. Własność 3 mówi o tym, że P_G jest zamknięty ze względu na prefiksy (czyli każdy prefiks każdego elementu też do niego należy). Łatwiej to wszystko ogarnąć patrząc na przykład.

Przykład 1. *Prosta gra:*

$$\begin{aligned} M_G &= \{a_1, a_2, b_1, b_2, b_3\} \\ \lambda_G &: a_1, a_2 \mapsto O, \quad b_1, b_2, b_3 \mapsto P \\ P_G &= \{\epsilon, a_1, a_1 b_1, a_2, a_2 b_2, a_2 b_3\} \end{aligned}$$

Przykład 1 pokazuje prostą grę, gdzie gracz ma do dyspozycji trzy ruchy, a przeciwnik tylko dwa. Zbiór P_G opisuje wszystkie możliwości, jakimi gra się może potoczyć. Jest to zapis drzewa, w którego korzeniu znajduje się ϵ, a krawędziami są ruchy gracza i przeciwnika.

2. Gry dla podstawowych typów

2.1. Liczby naturalne

Przeciwnik pyta o liczbę, a gracz może mu odpowiedzieć dowolną liczbą.

Definicja 2. *Gra $G_{\mathbb{N}}$ dla liczb naturalnych:*

$$\begin{aligned} M_{\mathbb{N}} &= \{q, 0, 1, 2, \ldots\} \\ \lambda_{\mathbb{N}}(q) &= O \\ \lambda_{\mathbb{N}}(n) &= P \quad \mathit{dla}\ n \in \{0, 1, 2, \ldots\} \end{aligned}$$

Jedyne poprawne rozgrywki to te postaci $q \cdot n$.

2.2. Krotki

Definicja 3. *Dla gier A i B możemy zdefiniować grę $A \times B$:*

$$\begin{aligned} M_{A\times B} &= M_A + M_B \\ \lambda_{A\times B}(x) &= \begin{cases} \lambda_A(x) & \textit{dla } x \in M_A \\ \lambda_B(x) & \textit{dla } x \in M_B \end{cases} \\ P_{A\times B} &= \{s \in M_{A\times B} \mid s \upharpoonright M_A \in P_A \wedge s \upharpoonright M_B \in P_B\} \end{aligned}$$

Innymi słowy są to dwie rozgrywki równoległe. Tylko przeciwnik może zmieniać grę.

2.3. Funkcje

Definicja 4. *Dla gier A i B możemy zdefiniować grę $A \Rightarrow B$:*

$$\begin{aligned} M_{A\Rightarrow B} &= M_A + M_B \\ \lambda_{A\Rightarrow B}(x) &= \begin{cases} \overline{\lambda}_A(x) & \textit{dla } x \in M_A \\ \lambda_B(x) & \textit{dla } x \in M_B \end{cases} \\ P_{A\Rightarrow B} &= \{s \in M_{A\times B} \mid s \upharpoonright M_A \in \overline{P}_A \wedge s \upharpoonright M_B \in P_B\} \end{aligned}$$

Gdzie $\overline{\lambda}_A$ to funkcja z odwróconym etykietowaniem (ruchy gracza stają się ruchami przeciwnika i na odwrót), a $\overline{P}_A$ to P_A z etykietowaniem $\overline{\lambda}_A$.

Są to dwie rozgrywki przeciwległe: w jednej jesteśmy graczem a w drugiej – przeciwnikiem. Tylko gracz może zmieniać grę.

3. Strategia

Strategią będziemy nazywali takie poddrzewo P_G, które w każdym wierzchołku ma tylko jeden ruch gracza.

Definicja 5. *Formalnie, deterministyczna strategia σ dla gry G, to taki niepusty podzbiór P_G, że:*

$$\forall s \in \sigma \quad |s| \textit{ jest parzysta,} \tag{4}$$

$$\epsilon \in \sigma, \tag{5}$$

$$\forall s \in P_G, \forall a, b \in M_G \quad sab \in \sigma \Rightarrow s \in \sigma, \tag{6}$$

$$\forall s \in \sigma, \forall a, b, c \in M_G \quad sab \in \sigma \wedge sac \in \sigma \Rightarrow b = c. \tag{7}$$

Własność 6 mówi nam, że strategia powinna być zamknięta na prefiksy, podobnie jak w przypadku P_G. Własność 7 nadaje strategii determinizm. Strategią dla gry z przykładu 1 mógłby być zbiór $\{\epsilon, a_1, a_1b_1, a_2, a_2b_3\}$.

Dzięki strategii będziemy zawsze mogli odpowiedzieć na każdy ruch przeciwnika i nasz ruch będzie jednoznacznie.

3.1. Strategia *copy-cat*

Strategia *copy-cat* jest jedną z podstawowych strategii w semantyce gier. Jest ona zdefiniowana dla gry $A \Rightarrow A$. Polega ona na tym, że kopiujemy ruchy przeciwnika z jednej gry do drugiej gry, jako nasze ruchy:

Ruch	Pierwsza gra	Druga gra	Czyj ruch
1		a_1	O
2	a_1		P
3	a_2		O
4		a_2	P
$\vdots$	$\vdots$	$\vdots$	$\vdots$

Ta strategia gwarantuje, że w jednej grze na pewno wygramy.

3.2. Program jako strategia

Program $x : A, y : B \vdash P : C$ można zinterpretować jako strategię gry $A \times B \Rightarrow C$. Zapis $x : A, y : B \vdash P : C$ oznacza, że na wejściu program dostaje x oraz y o typie odpowiednio A i B, a na wyjściu zwraca P typu C.

4. Semantyka Wyidealizowanego Algola

Pokażemy na przykładzie języka IA (*Idealized Algol*) w jaki sposób można zdefiniować semantykę za pomocą gier.

4.1. Składnia

Język IA to prosty język oparty na rachunku lambda z typami $\mathbb{N}$ i $\mathbb{B}$ odpowiednio dla liczb naturalnych i wartości logicznych. Dla instrukcji jest typ `comm` zawierający przypisanie $x := M$, złożenie $C_1; C_2$ oraz `skip` i lokalny blok `new x in M`. Zmienne są typu `var`, a napis $!x$ oznacza wartość zmiennej x.

4.2. Semantyka

Gry $\mathbb{N}$ i $\mathbb{B}$ definiujemy podobnie jak w rozdziale 2.1: jeden ruch przeciwnika q oraz ruchy gracza odpowiednio $1, 2, \ldots$ i T, F. Gra dla instrukcji jest podobna. Ma tylko dwa ruchy: `run` (przeciwnika) i `done` (gracza). Znacznie ciekawsza jest gra dla zmiennych. Mamy tutaj ruchy przeciwnika postaci `write(n)` i `read` (gdzie n jest liczbą naturalną), a ruchy gracza to `ok` i $1, 2, \ldots$.

Przykład 2. *Program przypisuje zmiennej x wartość y powiększoną o 1:*

$$
\begin{array}{ccc}
x : \text{var}, & y : \text{var} & \vdash x :=!y+1 : \quad \text{comm} \\
 & & \text{run} \\
 & \text{read} & \\
 & n & \\
\text{write}(n+1) & & \\
\text{ok} & & \\
 & & \text{done}
\end{array}
$$

Literatura

[1] S. Abramsky: *Semantics of Interactions*
[2] G. McCusker: *Game Semantics: an Overview*
[3] D. R. Ghica, G. McCusker: *The Regular-Language Semantics of Second-Ordered Idealized* ALGOL