Ponad-wielomianowy czas działania algorytmu *Strategy Improvement*

Michał Skrzypczak

21 maja 2009

Spis treści

Streszczenie

Dokument ten stanowi tłumaczenie pracy [1]. Zostaje wprowadzone pojęcie gry parzystości, oraz dokonana jest analiza algorytmów rozwiązujących takie gry. Głównym wynikiem (pochodzącym z tłumaczonej pracy) jest wskazanie rodziny gier parzystości, dla których algorytm *Strategy Improvement* działa w czasie wykładniczym.

1 Wprowadzenie

Gra parzystości to klasyczny przykład gry nieskończonej. Gra taka rozgrywa się na grafie skierowanym (V, E), gdzie wierzchołki są podzielone na dwa rozłączne podzbiory V_0, V_1. Dodatkowo każdy wierzchołek ma przypisany *rank*, będący liczbą naturalną. Gra rozpoczyna się w jakimś wybranym wierzchołku $v \in V$, następnie gracze wykonują ruchy wzdłuż krawędzi grafu. W danym momencie ruch wykonuje ten gracz, do którego ze zbiorów V_0, V_1 dany wierzchołek należy. Rozgrywka jest nieskończona, wygrywa ją gracz 0 wtw. najwyższy rank występujący nieskończenie wiele razy jest parzysty.

Poniżej wprowadzonych zostaje kilka notacji i spostrzeżeń związanych z grami parzystości.

- Dla dowolnych $v, u \in V$, będziemy pisać vEu, wtw. $(v, u) \in E$, czyli gdy istnieje krawędź z v do u.
- Funkcję przypisującą wierzchołkom grafu ich rank oznaczać będziemy $\Omega \colon V \to \mathbb{N}$.
- Przez *zysk* wierzchołka $v \in V$ (ozn. $\mathrm{rew}(v)$), oznaczać będziemy wartość $\Omega(V) \cdot (-1)^{\Omega(v)}$. Intuicyjnie im większy zysk danego wierzchołka, tym lepszy jest on z punktu widzenia gracza 0.
- Bez straty ogólności można zakładać, że z każdego wierzchołka w grze parzystości istnieje przynajmniej jedna krawędź wychodząca. Dzięki temu wystarczy rozpatrywać nieskończone rozgrywki.
- Bez straty ogólności można zakładać, że ranki przypisywane wierzchołkom grafu są parami różne.

Definicja 1.1. Strategią pozycyjną *gracza i nazywamy dowolną funkcję $\sigma \colon V_i \to V$, taką że $vE\sigma(v)$. Dla ustalonej strategii pozycyjnej σ, powiemy że rozgrywka* przebiega zgodnie z σ*, gdy w każdym wierzchołku $v \in V_i$, gracz i podejmuje decyzję by przejść do wierzchołka $\sigma(v)$.*

Łatwo wykazać, w oparciu o twierdzenie Martina [2], że gry parzystości są zdeterminowane, tzn. dla każdego wierzchołka początkowego, dokładnie jeden z graczy ma strategię wygrywającą. Co więcej, można wykazać że gracz posiadający strategię wygrywającą z danej pozycji, ma pozycyjną[1] strategię wygrywającą. Dowód tego faktu jest znacznie mniej oczywisty.

2 Rozwiązywanie gry parzystości

Przez *rozwiązanie* danej gry parzystości, rozumiemy znalezienie podziału zbioru wierzchołków na dwa rozłączne podzbiory W_0, W_1, będące zbiorami wierzchołków z których odpowiedni gracz ma strategię wygrywającą.

[1]Strategia pozycyjna to strategia w której decyzja dotycząca kolejnego ruchu zależy tylko od tego w którym wierzchołku znajdujemy się

Można stawiać to zadanie w następującej postaci: dla danej gry parzystości oraz wierzchołka początkowego v, odpowiedzieć który z graczy ma strategię wygrywającą z v. Oba problemy to dobrze postawione zadania algorytmiczne, gdyż rozpatrujemy tylko skończone grafy gry. I z dokładnością do czynnika wielomianowego mają tę samą złożoność.

Przykład 2.1. *Istnieje algorytm rozwiązujący gry parzystości, działający w czasie* $\mathcal{O}(|V| \cdot 2^{|E|})$.

Dowód. Ustalmy wierzchołek początkowy gry v. Zauważmy, że dla danych strategii pozycyjnych obu graczy, sprawdzenie który z graczy wygra, gdy obaj używać będą tych strategii, zajmuje czas liniowy ze względu na rozmiar grafu. Wystarczy wykonywać ruchy zgodne ze strategiami graczy, aż do pierwszej pętli, następnie sprawdzić jaki jest najwyższy rank występujący w ramach danej pętli.

Zawsze zachodzi dokładnie jedna z następujących możliwości.

- Gracz 0 posiada strategię pozycyjną σ, która wygrywa z każdą strategią pozycyjną gracza 1. Wtedy gracz 1 nie może mieć wygrywającej strategii pozycyjnej, więc nie ma w ogóle strategii wygrywającej z wierzchołka v. Więc gracz 0 ma strategię wygrywającą z wierzchołka v[2].
- Dla każdej strategii pozycyjnej gracza 0, istnieje strategia pozycyjna gracza 1, taka że gracz 1 wygrywa gdy obaj wybiorą odpowiednie strategie. Wtedy gracz 0 nie może mieć wygrywającej strategii pozycyjnej, więc posiada ją gracz 1.

Wobec tego wystarczy dla każdej strategii pozycyjnej σ_0 gracza 0, dla każdej strategii pozycyjnej σ_1 gracza 1, sprawdzić która ze strategii wygrywa z drugą. I w ten sposób rozstrzygnąć z którą z powyższych możliwości mamy do czynienia.

Strategii pozycyjnych gracza 0 jest co najwyżej $2^{|E_0|}$, gdzie E_i to zbiór krawędzi wychodzących z V_i. Dla każdej z nich trzeba sprawdzić nie więcej niż $2^{|E_1|}$ strategii pozycyjnych gracza 1. Więc w sumie sprawdzamy $2^{|E|}$ par strategii pozycyjnych i dla każdej pary wykonujemy akcję liniową ze względu na rozmiar grafu. Wobec tego algorytm działa w czasie $\mathcal{O}(|V| \cdot 2^{|E|})$. ■

Pytaniem otwartym jest czy daje się rozwiązywać gry parzystości w czasie wielomianowym ze względu na rozmiar grafu. Wiadomo natomiast, że należą one do klasy NP $\cap$ coNP. Odpowiednie redukcje są opisane w dalszej części pracy.

[2] A'priori nie musi być nią σ.

3 Definicje pomocnicze

Poniżej wprowadzimy definicje i pojęcia wykorzystywane w algorytmie *Strategy Improvement*.

Definicja 3.1. Ścieżką bez pętli *w grafie G nazywamy różnowartościowy ciąg wierzchołków grafu $\pi = \pi_0, \pi_1, \ldots, \pi_{k-1}$, o własności $\pi_i E \pi_{i+1}$ dla $i < k$. Długością takiej ścieżki (ozn. $|\pi|$) jest k. Zbiór wszystkich ścieżek bez pętli w danym grafie G, zaczynających się w danym wierzchołku $v \in G$, oznaczamy $\Pi_G(v)$.*

Definicja 3.2. *Wierzchołek v grafu G nazywamy* dominującym w cyklu*, jeśli należy on do pewnego cyklu w tym grafie i ma w ramach niego największy rank. Ściślej, jeśli istnieje ścieżka bez pętli π, zaczynająca się w v, o własności $\pi_{|\pi|-1} E v$, taka że $\Omega(v)$ jest maksymalnym występującym na niej rankiem. Zbiór wszystkich wierzchołków dominujących w cyklu w danym grafie, oznaczamy $\mathcal{C}_G$. Oczywiście odpowiednie ścieżki dla wierzchołków z tego zbioru mogą być różne.*

Zauważmy, że na każdym cyklu w dowolnym grafie G leży dokładnie jeden wierzchołek o największym ranku w ramach tego cyklu. Ponieważ założyliśmy, że z każdego wierzchołka wychodzi przynajmniej jedna krawędź, więc w każdym rozpatrywanym grafie istnieje przynajmniej jeden wierzchołek dominujący w cyklu.

Definicja 3.3. Wierzchołkiem wartościowym *nazywamy każdy wierzchołek $v \in V$, dla którego* $\mathrm{rew}(v) \geqslant 0$*. Innymi słowy jest to wierzchołek o parzystym ranku. Słowo wartościowy odnosi się (jak w całej tej pracy) do punktu widzenia gracza* 0*. Zbiór takich wierzchołków oznaczamy V_+. Pozostałe wierzchołki grafu są nie wartościowe dla* 0*, oznaczamy je V_-.*

Definicja 3.4. *Każda strategia pozycyjna σ dowolnego gracza i, dla ustalonego grafu gry G, wyznacza graf obcięty do danej strategii. Graf taki oznaczamy $G|_\sigma$. Powstaje on poprzez wycięcie z grafu G wszystkich krawędzi $v \to u$, spełniających warunek*

$$v \in V_i \wedge u \neq \sigma(v).$$

4 Wartościowania

Kluczowe pojęcia w algorytmach poprawiających strategie, w tym w algorytmie *Strategy Improvement*, to *wartościowanie* wierzchołków grafu. Chodzi o to by wierzchołkom grafu przypisywać ich ocenę w jakimś zbiorze uporządkowanym liniowo. Ocena taka mówi na ile dany wierzchołek jest wartościowy dla ustalonego gracza (zazwyczaj gracza 0). Wartościowanie to funkcja z wierzchołków grafu, w ich oceny.

W przypadku algorytmu *Strategy Improvement* oceny wierzchołków leżą w zbiorze $V \times 2^V \times |V|$. Każda ścieżka $\pi = v_0, v_1, \ldots, v_k$ definiuje ocenę wierzchołka v_0

$$\vartheta_\pi = (v_k, \{v_i | \mathrm{rew}(v_k) < \mathrm{rew}(v_i)\}, k+1).$$

Intuicyjnie należy myśleć, że każda ocena jest tej postaci i dodatkowo, że v_k jest wierzchołkiem dominującym w pętli w odpowiednim grafie. Jednak definicja tego nie wymusza.

W celu wprowadzenia porządku na ocenach wierzchołków musimy najpierw zdefiniować porządek na wszystkich podzbiorach V. Rozpatrzmy $M, N \subseteq V$. Powiemy, że $M < N$, w następujących dwóch przypadkach:

- $M \neq N$ i $\max_\Omega(M \triangle N) \in N \cap V_+$, czyli zbiory wierzchołków są różne, a wierzchołek o najwyższym ranku w ich różnicy symetrycznej jest wartościowy i leży w N,
- $M \neq N$ i $\max_\Omega(M \triangle N) \in M \cap V_-$, czyli zbiory wierzchołków są różne, a wierzchołek o najwyższym ranku w ich różnicy symetrycznej jest nie wartościowy i leży w M.

Zdefiniowana powyżej relacja $<$ jest porządkiem liniowym na podzbiorach V.

Teraz uporządkujemy liniowo oceny wierzchołków. Rozpatrzmy dwie oceny wierzchołków (u, M, e) i (v, N, f). Poniżej podana jest lista warunków. Powiemy, że $(u, M, e) < (v, N, f)$ wtw. jeden z warunków jest spełniony, a we wszystkich poprzednich występują odpowiednie równości.

- $\mathrm{rew}(u) < \mathrm{rew}(v)$, czyli porównujemy na ile wartościowy jest wierzchołek do którego zmierzamy,
- $M < N$, czyli jeśli wierzchołki do których zmierzamy są tej samej wartości, porównujemy sposób dojścia do nich,
- $(e < f \wedge u \in V_-)$, lub $(e > f \wedge u \in V_+)$, czyli gdy dochodzimy tak samo[3] do tych samych wierzchołków, patrzymy czy gracz 0 chce tam dojść szybko, czy wolałby odwlec ten moment.

Łatwo sprawdzić, że powyższa definicja wprowadza liniowy porządek na ocenach wierzchołków. Intuicyjnie porządek ten mówi na ile dobra jest dana ścieżka bez pętli wychodząca z danego wierzchołka. Przy czym cytując pewną koleżankę *detal tkwi w szczegółach.* Chodzi o to, że nie jest jasne (i niestety nie stanie się dużo bardziej jasne) dlaczego właśnie takie definicje są właściwe.

Definicja 4.1. Wartościowaniem grafu G *nazywamy funkcję* $\Xi \colon V \to V \times 2^V \times |V|$ *przypisującą wierzchołkom grafu jakieś ich oceny. Na wartościowaniach grafu wprowadzamy porządek częściowy* po współrzędnych.

[3] Z dokładnością do wierzchołków o niższych rankach niż docelowe.

Dla każdej strategii pozycyjnej σ gracza 0, określone jest w jakimś sensie *optymalne* (z punktu widzenia przeciwnika) wartościowanie. Zdefiniujmy je w następujący sposób

$$\Xi_\sigma(v) = \min_{<} \left\{ \vartheta_\pi : \pi \in \Pi_{G|_\sigma}(v) \wedge \pi_{|\pi|-1} \in \mathcal{C}_{G|_\sigma} \right\}.$$

Intuicja za tym stojąca jest następująca. Przy ustalonej strategii gracza 0 interesują nas tylko wybory gracza 1 w grafie $G|_\sigma$. Każda nieskończona ścieżka w tym grafie to potencjalny przebieg gry do jakiego gracz 1 może zmusić gracza 0 grającego σ. Analogicznie jak w przykładzie 2.1 wystarczy rozpatrywać strategie pozycyjne gracza 1. Czyli można przyjąć, że dalszy ciąg rozgrywki od ustalonego wierzchołka v będzie polegał na dojściu ścieżką bez pętli do jakiegoś wierzchołka u dominującego w cyklu, a następnie krążeniu po jego cyklu. Graczowi 1 pozostaje wybrać ten cykl. Powyżej składamy, że gracz 1 zrobi to optymalnie, czyli przede wszystkim minimalizując wartość $\mathrm{rew}(u)$. Gracz 1 ma strategię pozycyjną wygrywającą z σ wtedy i tylko wtedy gdy uda mu się uzyskać $\mathrm{rew}(u) < 0$.

W pracy [3] wykazane jest, że wartościowanie Ξ_σ można obliczyć w czasie wielomianowym ze względu na rozmiar grafu.

Oczywiście znając Ξ_σ możemy prosto obliczyć optymalną strategię gracza 1 przeciwko strategii 0. Zdefiniujmy ją następująco

$$\tau_\sigma(v) = \min_{\mathrm{rew}} \min_{\Xi_\sigma} \{u \in V : vEu\}.$$

Istotnym elementem powyższej definicji jest wybór minimum ze względu na Ξ_σ. Niestety dla różnych wierzchołków docelowych, możemy otrzymać dokładnie takie same wartości Ξ_σ. Wtedy wybieramy biorąc pod uwagę rew.

5 Poprawianie strategii

Dla danej strategii σ gracza 0, po obliczeniu Ξ_σ, możemy w oparciu o Ξ_σ polepszać strategię σ. Skutkuje to algorytmem iteracyjnym, w którym w każdym kroku liczymy wartość Ξ_σ dla aktualnie rozpatrywanej strategii σ, w oparciu o tę wartość poprawiamy σ do σ' i rozpoczynamy kolejny krok pętli. Najważniejszy składnik takiego algorytmu, to metoda za pomocą której znając σ i Ξ_σ obliczamy σ'. Aby określić tę metodę należy wpierw zdefiniować kolejne kilka pojęć.

Definicja 5.1. *Dla danej strategii σ gracza* 0*, przez* obszar poprawień *oznaczamy graf $A_{G,\sigma}$ powstały z G poprzez wykreślenie wszystkich krawędzi $v \to u$ spełniających następujący warunek*

$$v \in V_0 \wedge \Xi_\sigma(u) < \Xi_\sigma(\sigma(v)).$$

Czyli obszar poprawień, z dokładnością do krawędzi, jest takim samym grafem jak G. Krawędzi natomiast ma nie więcej niż G. Są tam wszystkie krawędzie należące do $G|_\sigma$. Dodatkowo są tam też te krawędzie, gdzie gracz 0 może podjąć decyzję nie gorszą niż σ przeciwko strategii wynikającej z Ξ_σ. I żadnej krawędzi poza tymi.

Definicja 5.2. *Powiemy, że strategia σ gracza* 0 *jest* poprawialna*, wtw. gdy istnieje wierzchołek $v \in V_0$, oraz wierzchołek $u \in V$, takie że vEu, oraz $\sigma(v) \neq u$, oraz $\Xi_\sigma(\sigma(v)) < \Xi_\sigma(u)$.*

Innymi słowy strategia jest poprawialna, gdy w pewnym wierzchołku grafu w którym gracz 0 podejmuje decyzję, może on ją podjąć inaczej niż dotychczas, zwiększając wartość Ξ_σ wierzchołka docelowego.

Można na to spojrzeć z punktu widzenia teorii gier. Dla ustalonej strategii σ gracza 0, wartościowanie Ξ_σ odpowiada najlepszej możliwej kontrstrategii gracza 1 oznaczonej τ_σ. I strategia jest poprawialna, wtw. gdy daje się grać lepiej przeciwko τ_σ, aniżeli grając σ. Jeśli znajdziemy parę σ, τ_σ, taką ze gracz 0 nie może grać lepiej przeciwko τ_σ, aniżeli grając właśnie σ, znaczy to że ta para odpowiada parze strategii będących w równowadze Nasha.

Definicja 5.3. Polityką poprawiania strategii *nazwiemy dowolną funkcję $\mathcal{I}_G \colon S_0(G) \to S_0(G)$, spełniającą*

- *dla każdego wierzchołka $v \in V_0$, para $(v, \mathcal{I}_G(\sigma)(v))$ jest krawędzią w $A_{G,\sigma}$,*
- *jeśli σ jest poprawialna, to $\Xi_\sigma \neq \Xi_{\mathcal{I}_G(\sigma)}$.*

Będziemy rozpatrywać polityki poprawiania strategii, które będą obliczalne w czasie wielomianowym ze względu na rozmiar grafu.

6 Algorytm

Twierdzenie 6.1. *Załóżmy, że strategia σ gracza* 0 *jest niepoprawialna. Wtedy gracz* 0 *ma strategię wygrywającą w wierzchołku v wtedy i tylko wtedy, gdy $\Xi_\sigma(v) = (u, \_, \_)$ i $\mathrm{rew}(u) \geqslant 0$.*

Dowód. Załóżmy, że strategia σ gracza 0 jest niepoprawialna. Rozpatrzmy dwa przypadki.

- Załóżmy, że $\Xi_\sigma(v) = (u, \_, \_)$ i $\mathrm{rew}(u) \geqslant 0$. Oznacza to, że najlepsza strategia pozycyjna jaką gracz 1 może przyjąć przeciwko strategii σ prowadzi do gry wygranej przez gracza 0. Czyli gracz 1 nie ma pozycyjnej strategii wygrywającej, więc posiada ją gracz 0.
- Rozumowanie w drugą stronę przeprowadzone jest w pracy [3].

∎

W pracy [3] powyższe twierdzenie wyrażane jest w mocniejszej postaci, mówiącej że strategia σ, lub strategia pochodząca od Ξ_σ są wygrywające w odpowiednim przypadku. Jednak do rozstrzygnięcia gry parzystości wystarczy nam twierdzenie w tej (prostszej) postaci.

W oparciu o powyższe twierdzenie, oraz wielomianowy algorytm obliczający Ξ_σ, łatwo wykazać, że rozwiązywanie gier parzystości należy do klasy NP. Rozpatrzmy mianowicie algorytm, który wczytuje opis gry parzystości, niedeterministycznie wybiera strategię pozycyjną σ gracza 0 (rozmiaru wielomianowego), a następnie oblicza Ξ_σ i sprawdza, kto wygrywa. Gracz 1 posiada strategię wygrywającą, wtw. gdy istnieje daje się wybrać wygrywającą σ. Podobnie, możemy wykazać, że problem ten leży w klasie $coNP$. Rozpatrzmy algorytm, który wczytuje opis gry parzystości, generuje niedeterministycznie strategie σ i dla każdej oblicza Ξ_σ. Jeśli dla każdej wygenerowanej strategii okazało się, że gracz 1 może wygrać, oznacza to że gracz 0 nie ma strategii wygrywającej.

Poniższe twierdzenie uzasadnia wartość wprowadzonego w tej pracy pojęcia oceny wierzchołka grafu.

Twierdzenie 6.2. *Dla dowolnej poprawialnej strategii σ gracza 0, zachodzi $\Xi_\sigma < \Xi_{\mathcal{I}_G(\sigma)}$, czyli w każdym punkcie odpowiednie oceny nie maleją, a w przynajmniej jednym rosną.*

Twierdzenie to jest udowodnione w pracy [3].

Dzięki dwóm powyższym twierdzeniom (oraz skończoności przestrzeni wartościowań), możemy sformułować poniższy algorytm rozwiązujący grę parzystości. Algorytm zwraca wartość logiczną, czy gracz 0 wygrywa w ustalonym wierzchołku v. Wykorzystywane w nim wartości to `i` oznaczające dowolną strategię gracza 0 oraz `I` oznaczające politykę poprawiania strategii.

```
let s : strategy for player 0 = i;
while s is improvable do
  s := I(s);
end;
let X : valuation of graph = Xi(s);
let (u, _, _) = X(v);
return (rew (u) >= 0);
```

Algorytm alokuje zmienną przyjmującą wartości w strategiach gracza 0. Początkowo zmienna ta przyjmuje wartość `i`. Następnie, w pętli, dopóki aktualnie rozpatrywana strategia jest poprawialna, zostaje ona poprawiona funkcją `I`. W każdym kroku pętli wartościowanie Ξ_s rośnie. W związku z tym kroków będzie wykonane co najwyżej skończenie wiele. Na koniec pętli obliczamy wartościowanie Ξ_s i oznaczamy je `X`. Pozostaje sprawdzić, czy funkcja rew na pierwszej współrzędnej wartości `X(v)` jest nieujemna i już wiemy kto wygrywa w danym wierzchołku v.

Zauważmy, że liczba kroków algorytmu jest potencjalnie wykładnicza ze względu na rozmiar grafu, gdyż taka jest długość najdłuższych możliwych łańcuchów w porządku częściowym wartościowań. Zauważmy też, że każdy krok algorytmu przebiega w czasie wielomianowym ze względu na rozmiar grafu.

Aby algorytm był kompletny pozostaje zdefiniować strategię początkową `i`, oraz politykę poprawiania strategii `I`. Rozpatrzmy niezwykle prostą strategię początkową

$$i(v) = \max_{\text{rew}}\{u : vEu\}.$$

Oczywiście w prawdziwych zastosowaniach można stosować bardziej złożone strategie początkowe, wykorzystujące różnorodne heurystyki by zmniejszyć średni czas działania algorytmu.

Jako politykę poprawiania strategii rozpatrywać będziemy *lokalną politykę poprawiania strategii*

$$I^l_G(\sigma)(v) = \max_{\text{rew}}\{u : vEu \wedge \forall_{w:vEw} \Xi_\sigma(w) \leqslant \Xi_\sigma(u)\}.$$

Alternatywą, w stosunku do powyższej polityki poprawiania strategii, jest *globalna polityka poprawiania strategii.* Jakkolwiek również w przypadku tej polityki daje się udowodnić poniższe fakty, my ograniczymy się do przypadku polityki lokalnej.

Oczywiście zarówno `i`, jak też `I` zdefiniowane powyżej można obliczać w czasie wielomianowym ze względu na rozmiar grafu.

7 Grafy

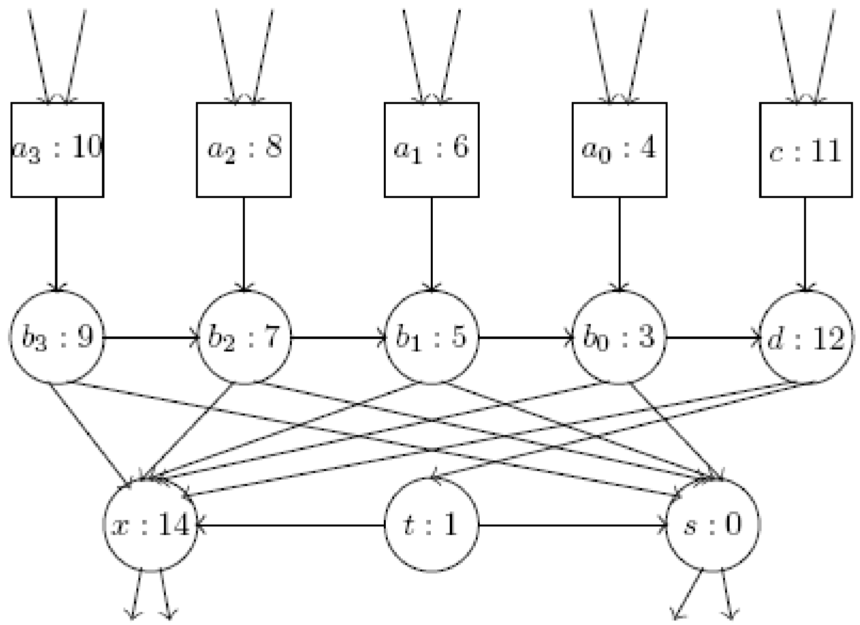
Zdefiniowany powyżej algorytm *Strategy Improvement* działa zazwyczaj szybko. Dobrze radzi sobie z grafami o regularnym kształcie, jak również z grafami losowymi. Dalsza część pracy ma na celu wskazanie rodziny grafów dla której algorytm ten działa w czasie wykładniczym w zależności od rozmiaru grafu.

Konstruowane grafy składają się z dwóch elementów (gadgetów). Pierwszy z nich to *linie zwalniające*, a drugi to *uparte cykle.* Poprzez sprytną kombinację tych dwóch składników daje się uzyskać grafy *trudne* dla algorytmu. Każdy taki graf będzie przegrywający dla gracza 0. Gracz 1 będzie w stanie dojść do wierzchołka oznaczanego $q \in V_1$ o ranku 1, z krawędzią $q \to q$. Trudny graf rozmiaru n składa się z jednej linii zwalniającej rozmiaru n, n upartych cykli, n pomocniczych struktur związanych z cyklami, oraz wierzchołków pomocniczych (w tym q). Na upartych cyklach takiego grafu skonstruowany zostaje licznik binarny, napędzany przez linię zwalniającą.

Zarówno pierwsza, jak i trzecia współrzędna ocen wierzchołków przyjmują wartości w zbiorach rozmiaru wielomianowego. Dlatego też współrzędne te nie są pomocne w wygenerowaniu długiego, malejącego łańcucha ocen wierzchołków. Poniższa konstrukcja w ogóle nie wykorzystuje tych współrzędnych, pierwsza współrzędna to zawsze wierzchołek q, a ostatnia dla każdego wierzchołka w grafie jest niezmienna. To co zmienia się w trakcie działania algorytmu i powoduje wykładniczy czas jego działania, to środkowa współrzędna, czyli zbiór istotnych wierzchołków na ścieżce do q. Ściślej, jest to zbiór tych wierzchołków na wybranej ścieżce do q, których $\mathrm{rew} > \mathrm{rew}(q) = -1$. Czyli zbiór wierzchołków o parzystym ranku. Zdefiniowany wcześniej porządek $<$ na zbiorach wierzchołków grafu sprowadza się w tym momencie do następującej formy: $M < N$ wtw. gdy wierzchołek o najwyższym ranku w ich różnicy symetrycznej leży w N.

7.1 Linie zwalniające

Linia zwalniająca rozmiaru $k + 1$, to graf składający się z trzech warstw. Poniżej znajduje się obrazek takiej linii rozmiaru 3.



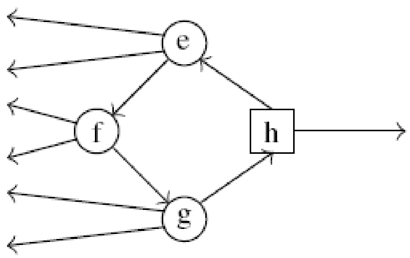
Zauważmy, że przy opisanej wyżej strategii początkowej σgracza 0 (grającego kółkami), σ mapuje wszystkie kółka powyższego grafu oprócz x i s w x. Załóżmy na razie, że ocena x jest stale większa niż ocena s. W takim przypadku po jednym kroku algorytmu, strategia zmienia się tylko o tyle że $\sigma(b_0) = d$. Po kolejnym kroku dodatkowo $\sigma(b_1) = b_0$. I tak dalej. Oczywiście za każdą taką zmianą odpowiedni wierzchołek z wiersza a zwiększa swoją ocenę.

Dodatkowo warto zauważyć, że kiedykolwiek wierzchołek s osiągnie wyższą ocenę aniżeli x, natychmiast wszystkie wierzchołki gracza 0 (oprócz

d) kierują się ku s. Taką sytuację nazywamy *zresetowaniem* linii. Jeśli potem znowu x zacznie być oceniany wyżej, cała zabawa zacznie się od początku. Wierzchołek t jest pomocniczy, w przypadku lokalnej polityki poprawiania strategii można go pominąć. Dzięki niemu ten sam graf będzie dobry dla globalnej polityki poprawiania strategii.

7.2 Uparte cykle

Uparty cykl składa się z czterech wierzchołków, każdy posiada jedną krawędź do środka i przynajmniej jedną na zewnątrz. Największy rank występujący z cyklu jest zawsze parzysty. Oto przykład upartego cyklu.



Cykl jest zamknięty jeśli aktualna strategia gracza 0 przypisuje $e \to f \to g \to h$. W takiej sytuacji gracz 1, licząc wartość Ξ_σ rozpatruje tylko ścieżki wychodzące z cyklu poprzez wierzchołek h. Z każdym cyklem występującym w grafie wiązać będziemy jeden bit, postawiony jeśli odpowiedni cykl jest zamknięty.

Łatwo wyobrazić sobie bieg algorytmu w którym określony uparty cykl zostaje szybko zamknięty, gracz 1 oblicza Ξ_σ w oparciu o krawędzie wychodzące z wierzchołka h i zmiany strategii w cyklu przestają zachodzić. Aby tego uniknąć i przedłużyć aktywność cyklu, należy zapewnić mu ciągłe zmiany wartościowań. W tym celu krawędzie wychodzące z cyklu łączy się z resztą grafu w taki sposób, by na przemian największą wartość miały następniki (nie należące do cyklu) wierzchołków $e, g, f, e, g, f, \ldots$. Dzięki temu stale utrzymywana jest sytuacja w której aktualna strategia gracza 0 wyprowadza dwie krawędzie poza cykl, a tylko jedną w ramach niego.

Efekt opisanych powyżej zmian wartościowań uzyskuje się poprzez połączenie cyklu z linią zwalniającą, odpowiednie wierzchołki cyklu do co trzecich wierzchołków a_i. W ten sposób, ciągłe poprawianie się linii zwalniającej będzie stale stymulować cykl do modyfikacji. W dalszej części pracy opisane jest ściślej, jak przebiegają te połączenia.

Zauważmy dodatkowo, że zamknięty uparty cykl jest nieczuły na modyfikacje opisanych powyżej dodatkowych krawędzi.

7.3 Cały graf

Opisywane grafy są złożone i składają się z licznych szczegółów. Natomiast ich ogólna idea jest dosyć klarowna. Połączona zostaje jedna linia zwalniająca, n upartych cykli, n struktur pomocniczych oraz cztery pomocnicze wierzchołki. k-ty uparty cykl jest podłączony (odpowiednio do co-trzecich) wierzchołków a_i dla $i < 3k$. Dodatkowo każdy z cykli ma swoją strukturę pomocniczą, w momencie zamknięcia danego cyklu, struktura pomocnicza powoduje otwarcie wszystkich cykli o mniejszych numerach, oraz reset linii zwalniającej.

Dzięki takiej konstrukcji, w miarę poprawiania linii zwalniającej coraz mniej otwartych cykli jest napędzanych zmianami wartości. Jako pierwszy zamknięty zostanie ten z otwartych cykli który posiada najmniejszy numer. Wtedy wyzwolona zostanie jego struktura pomocnicza, zamknięte cykle o mniejszych numerach się otworzą, a linia zwalniająca zostanie zresetowana.

Algorytm poprawiający strategie działa na takim grafie w rundach. Pojedyncza runda rozpoczyna się w momencie gdy linia zwalniająca jest zrestartowana i kończy się na kolejnym resecie. Podczas jednej takiej rundy wartość licznika binarnego reprezentowanego przez zamknięte (1) i otwarte (0) uparte cykle, zwiększa się o jeden. Koszt czasowy pojedynczej rundy zależy od pozycji najmłodszego otwartego cyklu, oszacować go można liniowo ze względu na rozmiar grafu. Ponieważ dopóki licznik się nie przepełni, dopóty kolejne rundy będą miały miejsce, więc cały bieg algorytmu wymaga czasu rzędu 2^n. Dodatkowo algorytm inicjuje początkowy stan licznika $0, 0, \ldots, 0, 1$ w czasie stałym, oraz po przepełnieniu licznika kończy pracę w czasie $\mathcal{O}(n)$.

Poniżej znajduje się obrazek prezentujący opisywany graf dla $n = 2$. Najbardziej lewe trzy kolumny to linia zwalniająca, dwa uparte cykle składają się z wierzchołków e_i, f_i, g_i, h_i, struktury pomocnicze indeksowane są k_i, l_i, z_i, n_i, a dodatkowe wierzchołki to w, y, p, q.

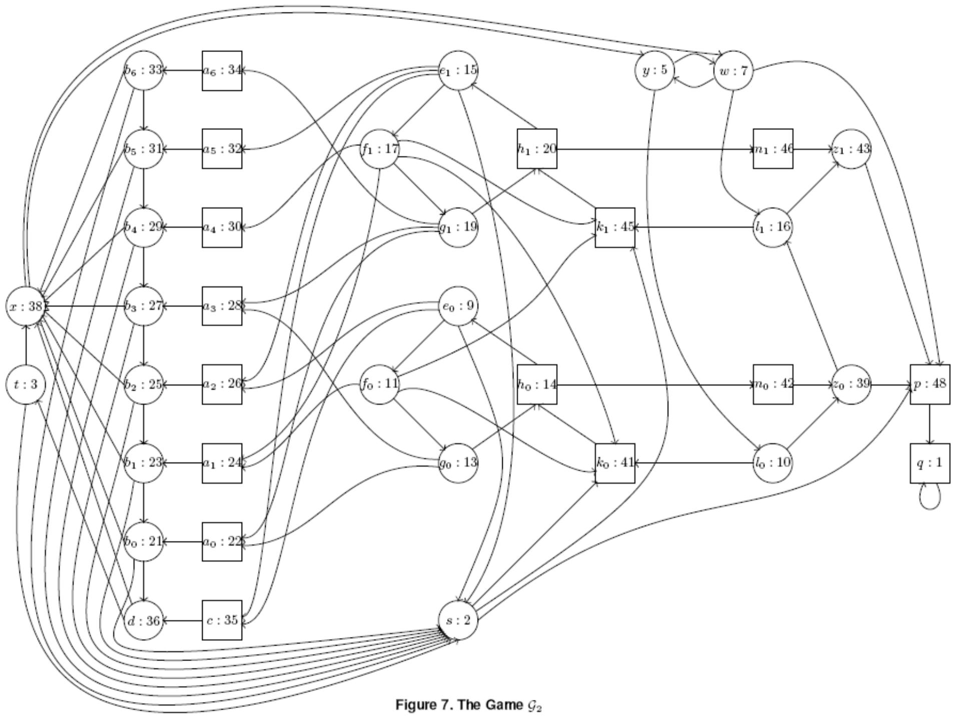


Figure 7. The Game $\mathcal{G}_2$

Literatura

[1] O. Friedmann, *A Super-Polynomial Lower Bound for the Parity Game Strategy Improvement Algorithm as We Know it*

[2] D. A. Martin, *Borel determinacy*, Ann. Math., 102:363–371, 1975.

[3] J. Vöge, M. Jurdzinski, *A discrete strategy improvement algorithm for solving parity games*, Proc. 12th Int. Conf. on Computer Aided Verification, CAV'00, tom 1855 of LNCS, strony 202–215. Springer, 2000.