Paper Soccer

Autor: Maciej Kasik 266463

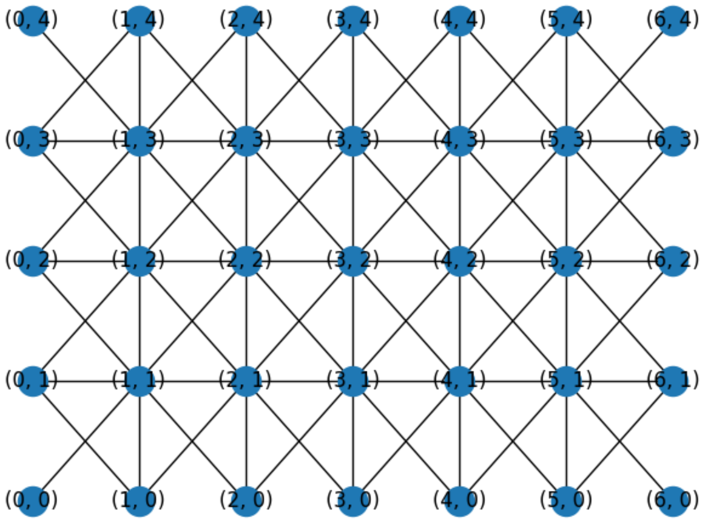
Krótki opis gry

Paper Soccer (lub papierowy hokej) to abstrakcyjna gra strategiczna rozgrywana na kwadratowej siatce, która przedstawia boisko do piłki nożnej lub hokeja. Grają w nią dwie osoby, które na zmianę wydłużają linię reprezentującą pozycję piłki, aż ta dotrze do jednego z dwóch pól bramkowych na siatce. Gra ta, tradycyjnie wykonywana na papierze i z użyciem ołówka, jest popularna w szkołach i czasem można ją znaleźć w niektórych czasopismach. Istnieje również wiele komputerowych implementacji tej gry. Pomimo prostych zasad, paper soccer oferuje różnorodne strategie i taktyki.

Zasady gry

1. Pole gry: Gra rozgrywana jest na kwadratowej lub prostokątnej siatce, która reprezentuje boisko do piłki nożnej lub hokeja. Siatka składa się z pól, które są połączone liniami.
2. Początek gry: Na początku gry ustawia się piłkę na środku siatki.
3. Ruchy: Gracze wykonują ruchy na zmianę. W swojej turze gracz wydłuża linię zaczynając od ostatniego punktu, w którym znajduje się piłka, i prowadzi ją do sąsiedniego pola na siatce. Piłka nie może przemieszczać się po brzegu boiska ani odcinkach, po których już wcześniej się przemieszczała, może jednak od nich się odbijać, tzn. jeśli w pozycji końcowej znajdował się przed wykonaniem ruchu koniec jakiegoś odcinka lub brzeg boiska, to po wykonaniu ruchu gracz wykonuje kolejny.
4. Bramki: Celem gry jest doprowadzenie piłki do jednego z dwóch pól bramkowych znajdujących się po przeciwnych stronach siatki.
5. Zasady ograniczeń: Piłka nie może opuścić pola gry ani przechodzić przez linie, które zostały już narysowane. Gracz nie może zakończyć swojego ruchu na polu, na którym już znajduje się linia lub piłka.
6. Koniec gry: Gra kończy się, gdy piłka dotrze do jednej z bramek. Gracz, który wprowadził piłkę do bramki, zdobywa punkt. Można też ustalić wcześniej określoną liczbę punktów do osiągnięcia przez graczy lub grać do ustalonego czasu.

Przykładowa plansza:



Reguła indeksowania wierzchołków:

$indeks\ =\ kolumna * iloscKolumn\ +\ wiersz$

$kolumna\ \in \{0,1,2,3,4,5,6\}$

$wiersz\ \in \{0,1,2,3,4\}$

Cel:

Naszym celem jest rozegranie całej partii gry, z dwoma zdefiniowanymi graczami minimalizując długość całej trasy i maksymalizując ilość tur.

Dane:

$l\ =\ 12$ – maksymalna z góry narzucona ilość tur

$d\ =\ 6$ – długość buforu związanego z ilością generowanych rozwiązań;

$H[h_{ij}]_{20x20}$ – macierz reprezentujaca graf z wszystkimi możliwymi ruchami.

$h_{ij} - \begin{cases} 1 - \textit{jeżeli istnieje połączenie pomiędzy wierzchołkami} \\ 0 - \textit{w przeciwnym wypadku} \end{cases}$

$g\ =\ 10$ – indeks wierzchołka początku naszego ruchu

$b_0\ =\ 1$ – indeks zakończenia całego ruchu (bramka gracza numer zero)

b_1 = 19 – indeks zakończenia całego ruchu (bramka gracza numer jeden)

i, j – indeks wierzchołka

$i, j \in \{0,1,2, \dots ,19\}$

$k - \textit{indeks tury}$

$k - \{0,1,2, \dots, l\}$
$n - indeks\ odwiedzonego\ wierzchołka\ w\ trasie$
$n - \{0,1,2, \dots, d\}$

Zmienne decyzyjne:

$X[x_{i,j}]_{20x20}$ – macierz przechowywujaca wszystkie odwiedzone już połączenia

$$x_{i,j} - \begin{cases} 1 - jeżeli\ istnieje\ połączenie\ pomiędzy\ wierzchołkami \\ 0 - w\ przeciwnym\ wypadku \end{cases}$$

$G[g_{k,i,j}]_{lx20x20}$ macierz przechowywujaca wszystkie odwiedzone już połączenia w trasie w każdej turze od końca nowego ruchu do bramki gracza numer zero

$$g_{k,i,j} - \begin{cases} 1 - jeżeli\ istnieje\ połączenie\ pomiędzy\ wierzchołkami \\ 0 - w\ przeciwnym\ wypadku \end{cases}$$

$V[v_{k,i,j}]_{lx20x20}$ – macierz przechowywujący wszystkie odwiedzone już połączenia w trasie w każdej turze od końca nowego ruchu do bramki gracza numer zero

$$v_{k,i,j} - \begin{cases} 1 - jeżeli\ istnieje\ połączenie\ pomiędzy\ wierzchołkami \\ 0 - w\ przeciwnym\ wypadku \end{cases}$$

$Z[z_{k,n}]_{lxd}$ bufor przechowywujący najkrotsza trase od bramki gracza numer zero do końcowego punktu trasy w każdej turze
$z_{k,n}$ – indeks odwiedzonego wierzchołka w trasie od bramki gracza numer zero

$W[w_{k,n}]_{lxd}$ – najkrótsza trasa od bramki gracza numer zero do końcowego punktu trasy w każdej turze po wyzerowaniu zbędnych rozwiązań
$w_{k,n}$ – indeks odwiedzonego wierzchołka w trasie od bramki gracza numer zero po wyzerowaniu zbędnych rozwiązań

$A[a_{k,n}]_{lxd}$ – macierz ostatnich indeksów trasy od bramki gracza numer zero do końcowego wierzchołka w każdej turze.
$a_{k,n}$ - ostatni indeks trasy od bramki gracza numer zero do końcowego wierzchołka w każdej turze

$C[c_{k,n}]_{lxd}$ – wyzerowane ostatnie indeksy trasy od bramki gracza numer zero do końcowego wierzchołka w każdej turze po uwzględnieniu komu przypada dana tura
$c_{k,n}$ - ostatni indeks trasy od bramki gracza numer zero do końcowego wierzchołka w każdej turze po uwzględnieniu komu przypada dana tura

$E[e_{k,n}]_{lxd}$ bufor przechowywujący najkrotsza trase od bramki gracza numer jeden do końcowego punktu trasy w każdej turze
$e_{k,n}$ – indeks odwiedzonego wierzchołka w trasie od bramki gracza numer jeden

$Q[q_{k,n}]_{lxd}$ – najkrótsza trasa od bramki gracza numer jeden do końcowego punktu trasy w każdej turze po wyzerowaniu zbędnych rozwiązań
$q_{k,n}$ – indeks odwiedzonego wierzchołka w trasie od bramki gracza numer jeden po wyzerowaniu zbędnych rozwiązań

$U[u_{k,n}]_{lxd}$ – macierz ostatnich indeksów trasy od bramki gracza numer jeden do końcowego wierzchołka w każdej turze
$u_{k,n}$ - ostatni indeks trasy od bramki gracza numer jeden do końcowego wierzchołka w każdej turze

$M[m_{k,n}]_{lxd}$ – wyzerowane ostatnie indeksy trasy od bramki gracza numer jeden do końcowego wierzchołka w każdej turze po uwzględnieniu komu przypada dana tura.
$m_{k,n}$- ostatni indeks trasy od bramki gracza numer jeden do końcowego wierzchołka w każdej turze po uwzględnieniu komu przypada dana tura

$\sigma[\sigma_{k,n}]_{lxd}$ – wektor pozwalajacy wyzerować niepotrzebne wartości w najkrótszej trasie do punktu w każdej turze

$$\sigma_{k,n} - \begin{cases} 1 - jeżeli\ wierzchołek\ nie\ powinien\ zostać\ wyzerowany \\ 0 - w\ przeciwnym\ wypadku \end{cases}$$

$\tau[\tau_{k,n}]_{lxd}$ – zmienna pomocnicza pozwalajaca wyzerować niepotrzebne wartości w najkrótszej trasie od bramki gracza zero do punktu w każdej turze

$$\tau_{k,n} - \begin{cases} 1 - \textit{jeżeli wierzchołek nie powinien zostać wyzerowany} \\ 0 - \textit{w przeciwnym wypadku} \end{cases}$$

$\delta[\delta_{k,n}]_{lxd}$ – zmienna pomocnicza pozwalajaca wyzerować niepotrzebne wartości w najkrótszej trasie od bramki gracza jeden do punktu w każdej turze.

$$\delta_{k,n} - \begin{cases} 1 - \textit{jeżeli wierzchołek nie powinien zostać wyzerowany} \\ 0 - \textit{w przeciwnym wypadku} \end{cases}$$

$\pi[\pi_{k,n}]$ – bufor przechowywujący trasę w każdej turze
$\pi_{k,n}$ – indeks kolejnego wierzchołka w danej turze.

$\mu[\mu_{k,n}]$ – macierz indeksów wierzchołków odwiedzonych w i-tej turze po wyzerowaniu
$\mu_{k,n}$ - indeks wierzchołka odwiedzonego w i-tej turze po wyzerowaniu

$S[s_k]$ -wektor przechowywujący liczbę odwiedzonych wierzchołków w i-tej turze

s_k - liczba odwiedzonych wierzchołków w i-tej turze

r -liczba wszystkich tur (indeks ostatniej tury)

$\gamma[\gamma_k]$ -wektor określający czyja jest tura

$$\gamma_k - \begin{cases} 1 - \textit{jeżeli tura należy do gracza numer jeden} \\ 0 - \textit{jeżeli tura należy do gracza numer zero} \end{cases}$$

$\rho[\rho_{k,i}]$ – macierz określająca czy dany wierzchołek został odwiedzony w danej turze

$$\rho_{k,i} - \begin{cases} 1 - \textit{jeżeli wierzchołek został odwiedzony} \\ 0 - \textit{w przeciwnym wypadku} \end{cases}$$

$\alpha[\alpha_{k,i}]$ – macierz przechowywująca ilość odwiedzin danego wierzchołka na przestrzeni tur

$\alpha_{k,i}$ – suma odwiedzin danego wierzchołka w k i w k-poprzednich turach.

$\beta[\beta_{k,i}]$ – macierz przechowywująca czy wierzchołek w trasie w turze był już odwiedzany wcześniej

$$\beta_{k,i} - \begin{cases} 1 - \textit{jeżeli wierzchołek z trasy w turze został już wcześniej odwiedzony} \\ 0 - \textit{w przeciwnym wypadku} \end{cases}$$

Funkcja celu:

$A(T) = \sum_k^l s_k$ - minimalizujemy sumę ostatnich indeksów trasy

$B(M) = \sum_k^l m_k$ - minimalizujemy sumę ostatnich indeksów trasy od bramki gracza numer zero

$C(Q) = \sum_k^l q_k$- minimalizujemy sumę ostatnich indeksów trasy od bramki gracza numer jeden

$D(r) = \infty - r$- maksymalizujemy ilość tur

$$G(T, M, Q, r) = 0.25 * A(T) + 0.25 * B(M) + 0.25 * C(Q) + 0.25 * D(r)$$

Ograniczenia:

$\gamma_0 = 1$ - ustawiam założenie, że pierwsza tura należy do pierwszego gracza.

$\forall_{o=n-1}^{n} \ \gamma_{o+1} = 1 - \gamma_o$ - tury mają być naprzemiennie

$\mu_0 = 1$ - w pierwszym ruchu gracz może wykonać maksymalnie jeden ruch.

$\mu_1 = 1$ – w drugim ruchu gracz może wykonać maksymalnie jeden ruch;

$\pi_{0,0} = g$ – początek naszego całego ruchu na przestrzeni wszystkich tur musi zacząć się w początkowym punkcie

$r \leq l$ – ilość tur musi być mniejsza niż maksymalna ilość tur

$r \geq 1$ – liczba wszystkich tur ma być większa od zera

$\forall_k \ s_k = 1$ – tymczasowe założenie (nie wiem jak zrobić dynamiczny ruch w turze)

$\forall_k \ s_k > 0$ - ostatni indeks trasy w każdej turze musi być większy od 0 (gracz musi wykonać jakiś ruch)

$\forall_{o=n-1}^{k,n} \ h_{\pi_{k,o},\pi_{k,o+1}} ! = 0$ -połączenie w grafie z wszystkimi dostępnymi ruchami musi istnieć

$\forall_{o=k-1}^{k} \ \pi_{o+1,0} = \pi_{o,s_o}$ – początkowy wierzchołek nowej tury musi być końcowym wierzchołkiem poprzedniej.

$\forall_k \ z_{k,0} = \pi_{k,s_k}$ - początkowy punkt w trasie od bramki gracza zero jest punktem końcowym w danej turze

$\forall_k \ z_{k,a_k} = b_0$ - końcowy punkt w trasie od bramki gracza zero jest indeksem bramki

$\forall_{k,n} \ h_{z_{k,n},z_{k,n+1}} ! = 0$ - połączenie w trasie od bramki gracza zero w grafie z wszystkimi dostępnymi ruchami musi istnieć

$\forall_{k,n} \ c_{n,} = \tau_{k,n} * z_{k,n}$ – wyzerowana trasa od bramki gracza zero do końcowego punktu w danej turze

$\forall_{k,n} \ (1 - \gamma_k) * e_{k,n} \ ! = b_1$ - gracz numer zero nie może wykonać ruchu do bramki gracza jeden

$\forall_{k,n} \ \gamma_k * e_{k,n} \ ! = b_0$ - gracz numer jeden nie może wykonać ruchu do bramki gracza zero

$\forall_{k,n} \ \mu_{k,n} = \sigma_{k,n} * \pi_{k,n}$ - zeruje niepotrzebne rozwiązania

$\forall_{o=n-1}^{k,n} \ x_{\mu_{k,n},\mu_{k,n+1}} = (\sum_f^l \sum_{y=r-1}^{d-1} {}_r (\mu_{k,o} = \mu_{f,y} \wedge \mu_{k,o+1} = \mu_{f,y+1}))$ - cała trasa w macierzy ma być równa sumie wszystkich istniejących krawędzi

$\forall_{o=n+1}^{k,n} \ x_{\mu_{k,n},\mu_{k,n-1}} = (\sum_f^l \sum_{y=r+1}^{d-1} {}_r (\mu_{k,o} = \mu_{f,y} \wedge \mu_{k,o-1} = \mu_{f,y-1}))$ - cała trasa w macierzy ma być równa sumie wszystkich istniejących krawędzi

$\forall_{o=n-1}^{k,n} \ g_{w_{k,o},w_{k,o+1}} = \sum_{r=f-1}^{d-1} {}_f \ (w_{k,o} = w_{k,r} \wedge w_{k,o+1} = w_{i,r+1})$ - cała trasa od bramki zero w macierzy ma być równa sumie wszystkich istniejących krawędzi

$\forall_{o=n+1}^{k,n} \ g_{w_{k,o},w_{k,o-1}} = \sum_{r=f+1}^{d-1} {}_f \ (w_{k,o} = w_{k,r} \wedge w_{k,o-1} = w_{k,r-1}))$ - cała trasa od bramki zero w macierzy ma być równa sumie wszystkich istniejących krawędzi

$\forall_{\substack{k,n \\ o=n-1}} v_{q_{k,o},q_{k,o+1}} = \sum_{r=f-1}^{d-1}{}_{f} (\mathrm{q}_{k,o} = \mathrm{q}_{k,r} \wedge \mathrm{q}_{k,o+1} = \mathrm{q}_{k,r+1})$- cała trasa od bramki jeden ma być równa sumie wszystkich istniejących krawędzi

$\forall_{\substack{k,n \\ o=n+1}} v_{q_{k,o},q_{k,o-1}} = \sum_{r=f+1}^{d-1}{}_{f} (\mathrm{q}_{k,o} = \mathrm{q}_{k,r} \wedge \mathrm{q}_{k,o-1} = \mathrm{q}_{k,r-1})$ - cała trasa od bramki jeden ma być równa sumie wszystkich istniejących krawędzi

$\forall_{\substack{y\in[1,20],t\in[1,20] \\ y,t\in N}} x_{y,t} \leq 1$- (ominięcie pozycji 00 w macierzy) połączenie może istnieć tylko raz.

$\forall_{\substack{y\in[1,20] \\ y\in N}} x_{y,0} \leq 1$-(ominięcie pozycji 00 w macierzy) połączenie może istnieć tylko raz.

$\forall_{\substack{y\in[1,20] \\ y\in N}} x_{0,y} \leq 1$- (ominięcie pozycji 00 w macierzy) połączenie może istnieć tylko raz.

$\forall_{\substack{y\in[1,20],t\in[1,20] \\ y,t\in N}} (x_{y,t} + x_{t,y}) \leq 1$-(ominięcie pozycji 00 w macierzy) ograniczenie związane ze sposobem zapisu grafu w postaci macierzy sąsiedztwa (graf nieskierowany) – może istnieć maksymalnie jedno połączenie pomiędzy wierzchołkami.

$\forall_{\substack{y\in[1,20] \\ y\in N}} (x_{y,0} + x_{0,y}) \leq 1$ --(ominięcie pozycji 00 w macierzy) ograniczenie związane ze sposobem zapisu grafu w postaci macierzy sąsiedztwa (graf nieskierowany) – może istnieć maksymalnie jedno połączenie pomiędzy wierzchołkami.

$\forall_{\substack{k \\ y\in[1,20],t\in[1,20] \\ y,t\in N}} \mathrm{g}_{k,y,t} \leq 1$ -(ominięcie pozycji 00 w macierzy) połączenie może istnieć tylko raz.

$\forall_{\substack{k \\ y\in[1,20] \\ y,t\in N}} \mathrm{g}_{k,y,0} \leq 1$ - (ominięcie pozycji 00 w macierzy) krawędź może istnieć tylko raz.

$\forall_{\substack{k \\ y\in[1,20] \\ y,t\in N}} \mathrm{g}_{k,0,y} \leq 1$- (ominięcie pozycji 00 w macierzy) krawędź może istnieć tylko raz.

$\forall_{\substack{k \\ y\in[1,20],t\in[1,20] \\ y,t\in N}} (g_{k,y,t} + \mathrm{g}_{k,t,y}) \leq 1$ –(ominięcie pozycji 00 w macierzy) ograniczenie związane ze sposobem zapisu grafu w postaci macierzy sąsiedztwa (graf nieskierowany) – może istnieć maksymalnie jedno połączenie pomiędzy wierzchołkami.

$\forall_{\substack{k \\ y\in[1,20] \\ y,t\in N}} (\mathrm{g}_{k,y,0} + \mathrm{g}_{k,0,y}) \leq 1$- (ominięcie pozycji 00 w macierzy) ograniczenie związane ze sposobem zapisu grafu w postaci macierzy sąsiedztwa (graf nieskierowany) – może istnieć maksymalnie jedno połączenie pomiędzy wierzchołkami.

$\forall_{\substack{k \\ y\in[1,20],t\in[1,20] \\ y,t\in N}} v_{k,y,t} \leq 1$- (ominięcie pozycji 00 w macierzy) ograniczenie związane ze sposobem zapisu grafu w postaci macierzy sąsiedztwa (graf nieskierowany) – może istnieć maksymalnie jedno połączenie pomiędzy wierzchołkami.

$\forall_{\substack{k \\ y\in[1,20] \\ y,t\in N}} v_{k,y,0} \leq 1$- (ominięcie pozycji 00 w macierzy) krawędź może istnieć tylko raz.

$\forall_{\substack{k \\ y\in[1,20] \\ y,t\in N}} \mathrm{v}_{k,0,y} \leq 1$- (ominięcie pozycji 00 w macierzy) krawędź może istnieć tylko raz.

$\forall_{\substack{k \\ y\in[1,20],t\in[1,20] \\ y,t\in N}} (\mathrm{v}_{k,y,t} + \mathrm{v}_{k,t,y}) \leq 1 -$ (ominięcie pozycji 00 w macierzy) ograniczenie związane ze sposobem zapisu grafu w postaci macierzy sąsiedztwa (graf nieskierowany) – może istnieć maksymalnie jedno połączenie pomiędzy wierzchołkami.

$\forall_{\substack{k \\ y\in[1,20] \\ y,t\in N}} (v_{k,y,0} + \mathrm{v}_{k,0,y}) \leq 1$- (ominięcie pozycji 00 w macierzy) ograniczenie związane ze sposobem zapisu grafu w postaci macierzy sąsiedztwa (graf nieskierowany) – może istnieć maksymalnie jedno połączenie pomiędzy wierzchołkami.

$\forall_k\ \mathrm{s}_k \leq d$ - ostatni indeks trasy w turze musi być mniejszy od długości trasy

$\forall_k\ \mathrm{a}_k \leq d$ – ostatni indeks trasy od bramki gracza zero w turze musi być mniejszy od długości trasy

$\forall_k\ \mathrm{u}_k \leq d$ - ostatni indeks trasy od bramki gracza jeden w turze musi być mniejszy od długości trasy

$\forall_{k,n}\ \sigma_{\ k,n} = (k \leq r) \wedge (n \leq s_k)$ - ograniczenie wskazujące które elementy należy wyzerować w trasie

$\forall_{k,n}\ \tau_{\ k,n} = (c_i = 0) \wedge (n \leq a_k)$ - ograniczenie wskazujące które elementy należy wyzerować w trasie od bramki gracza zero

$\forall_{k,n}\ \delta_{\ k,n} = (c_i = 1) \wedge (n \leq u_k)$ - ograniczenie wskazujące które elementy należy wyzerować w trasie od bramki gracza jeden

$\forall_{k,n}\ \rho_{\ k,\mu_{k,n}} = (k \leq r) \wedge (n \leq s_k)$ - ograniczenie określające czy dany wierzchołek został odwiedzony w danej turze

$\forall_{k,i}\ \beta_{k,i} = (\sigma_{k,i} * \alpha_{k,\mu_{k,i}}) \geq 1$- ograniczenie określające czy element trasy w turze był już wcześniej odwiedzany

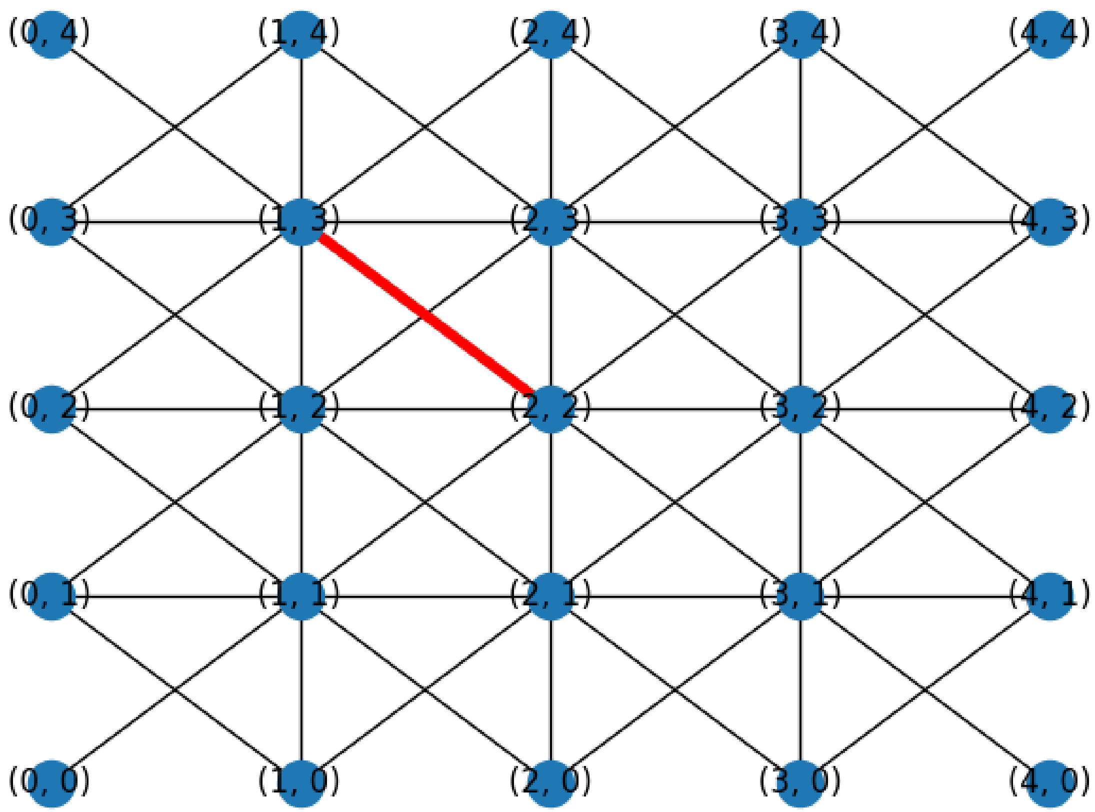
$\forall_{k,i}\ \beta_{k,i} = i < s_k$ - ograniczenie określające ilość możliwych do wykonania ruchów w danej turze

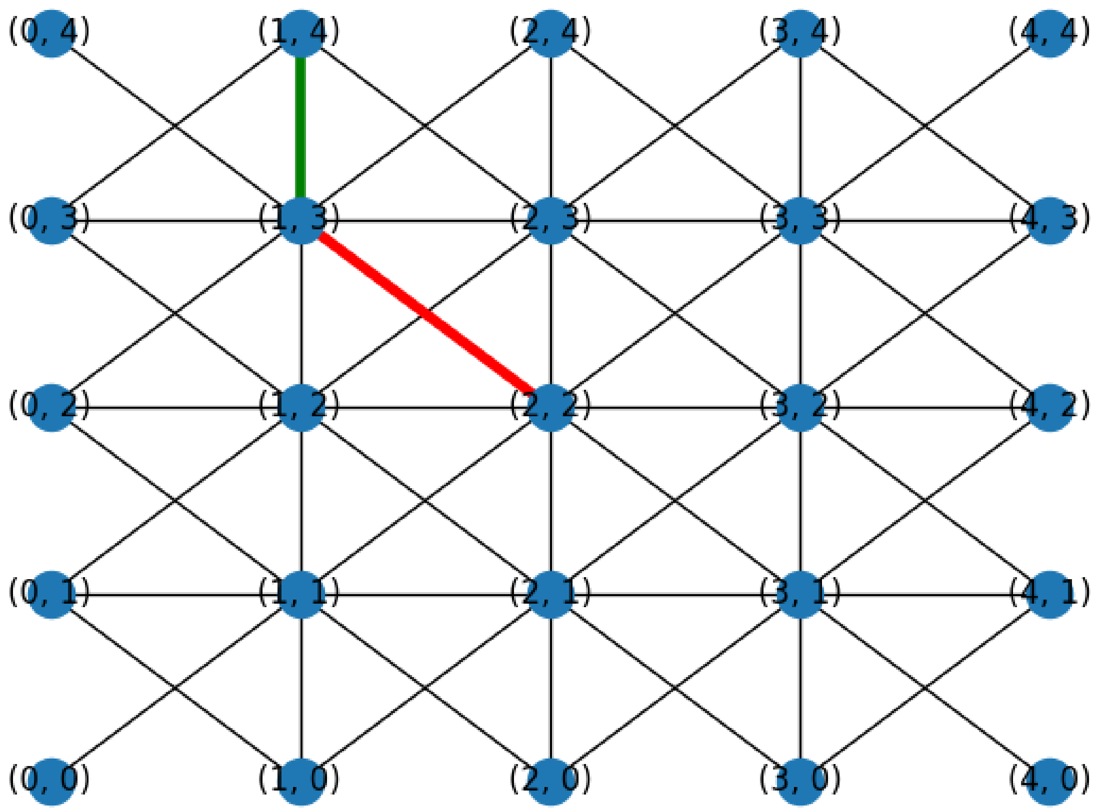
Przykładowa rozgrywka rozegrana przez solver:

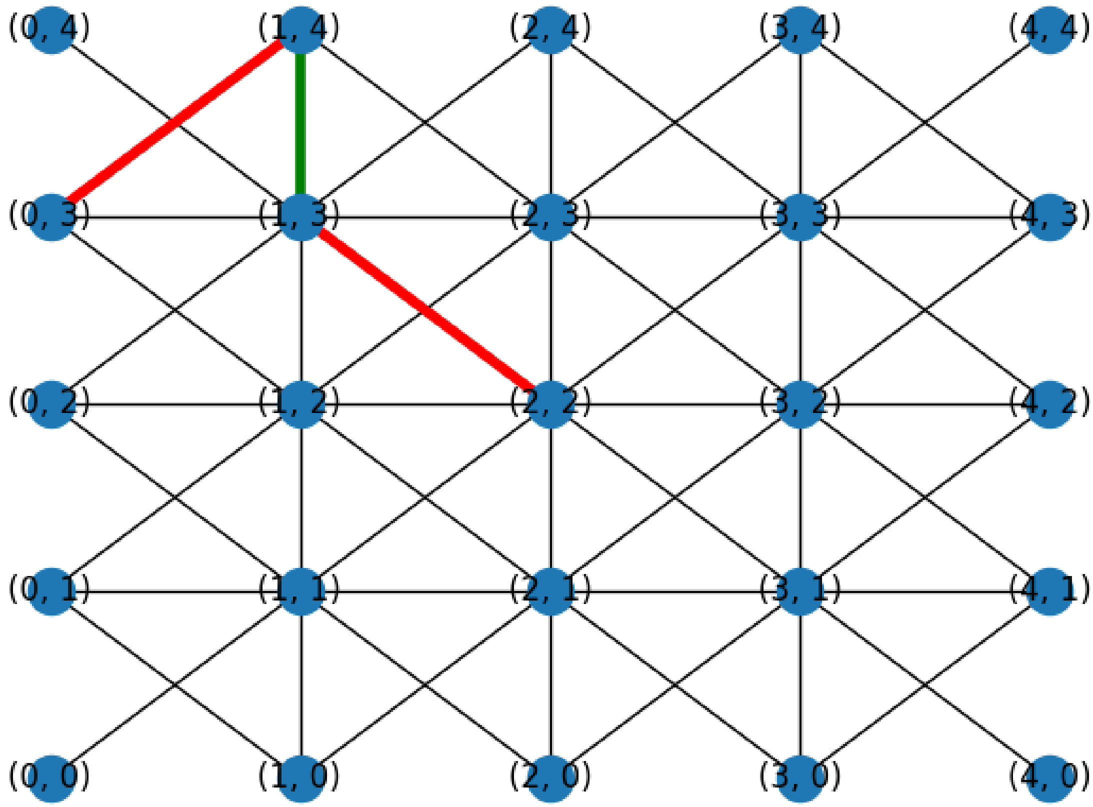
Czerwony kolor jest reprezentacją gracza gracza numer jeden.

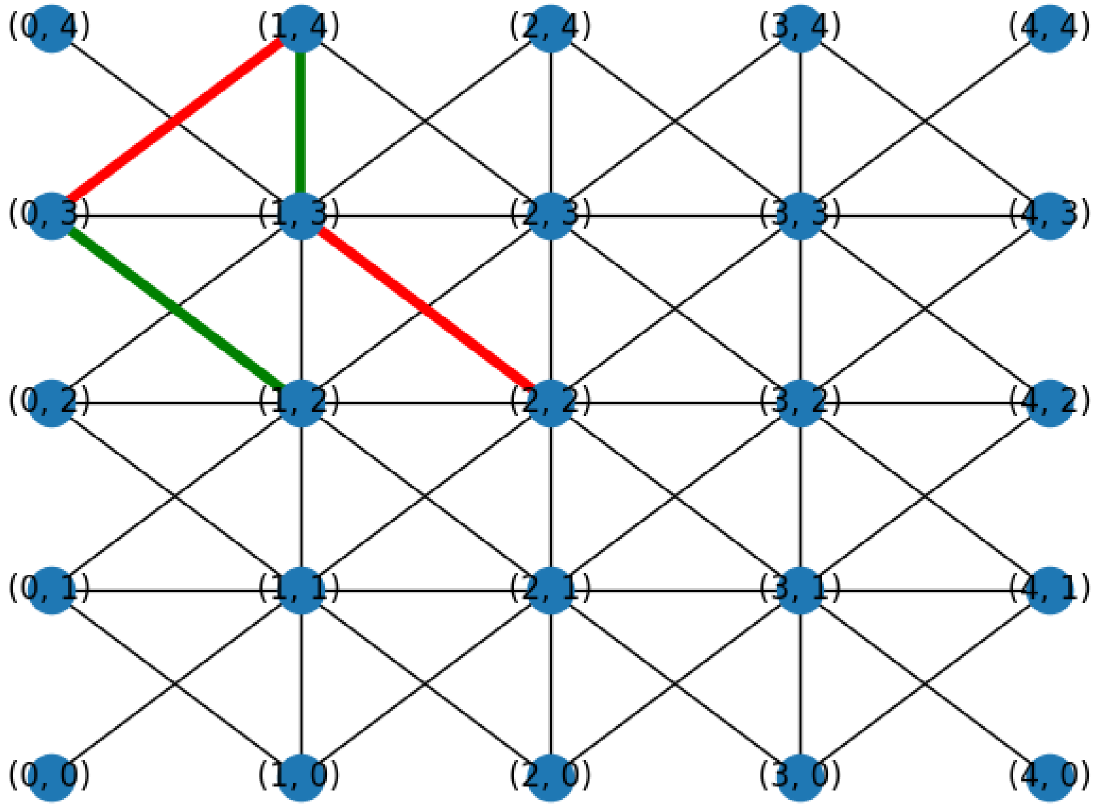
Zielony kolor jest reprezentacją gracza numer zero.

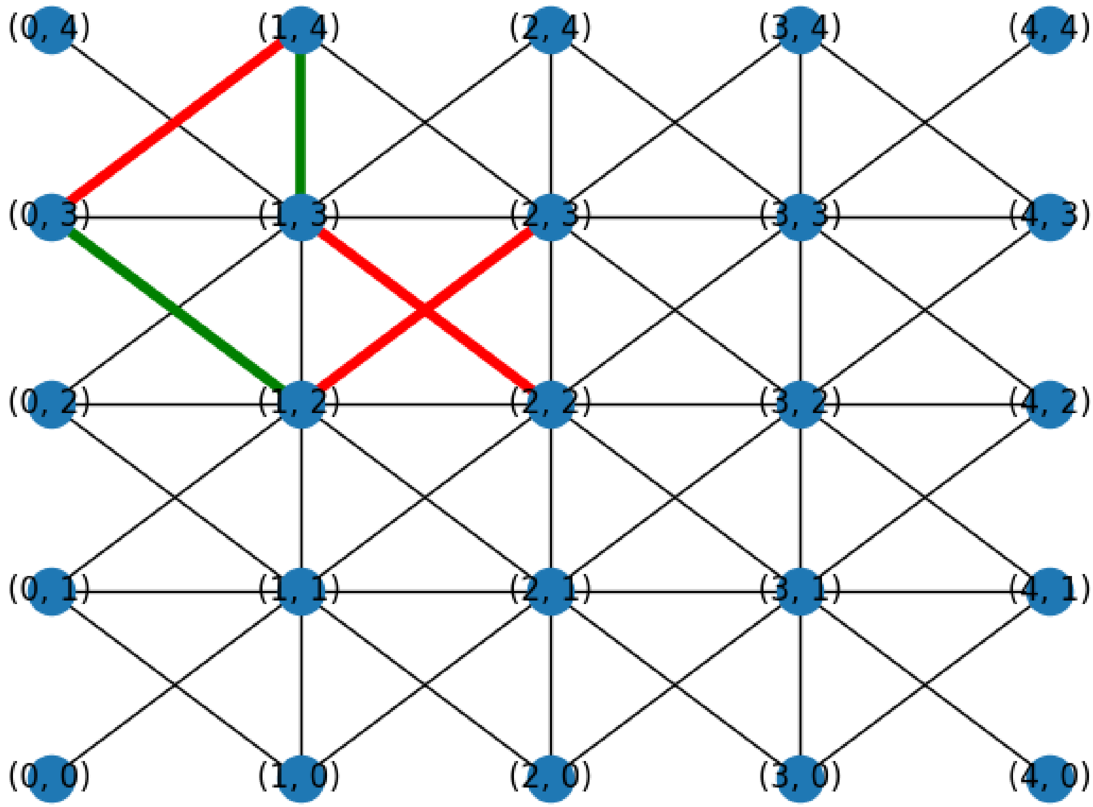
Początkowa tura należy do gracza numer jeden.

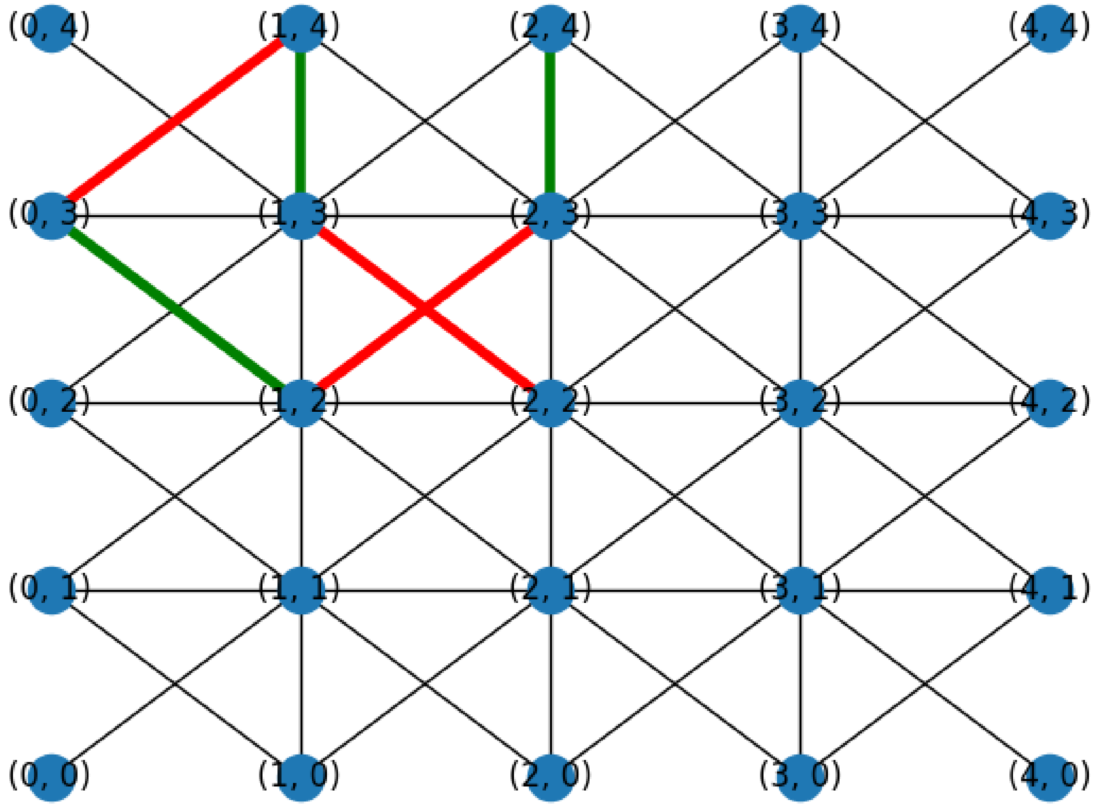

(0, 4)
(1, 4)
(2, 4)
(3, 4)
(4, 4)
(0, 3)
(1, 3)
(2, 3)
(3, 3)
(4, 3)
(0, 2)
(1, 2)
(2, 2)
(3, 2)
(4, 2)
(0, 1)
(1, 1)
(2, 1)
(3, 1)
(4, 1)
(0, 0)
(1, 0)
(2, 0)
(3, 0)
(4, 0)

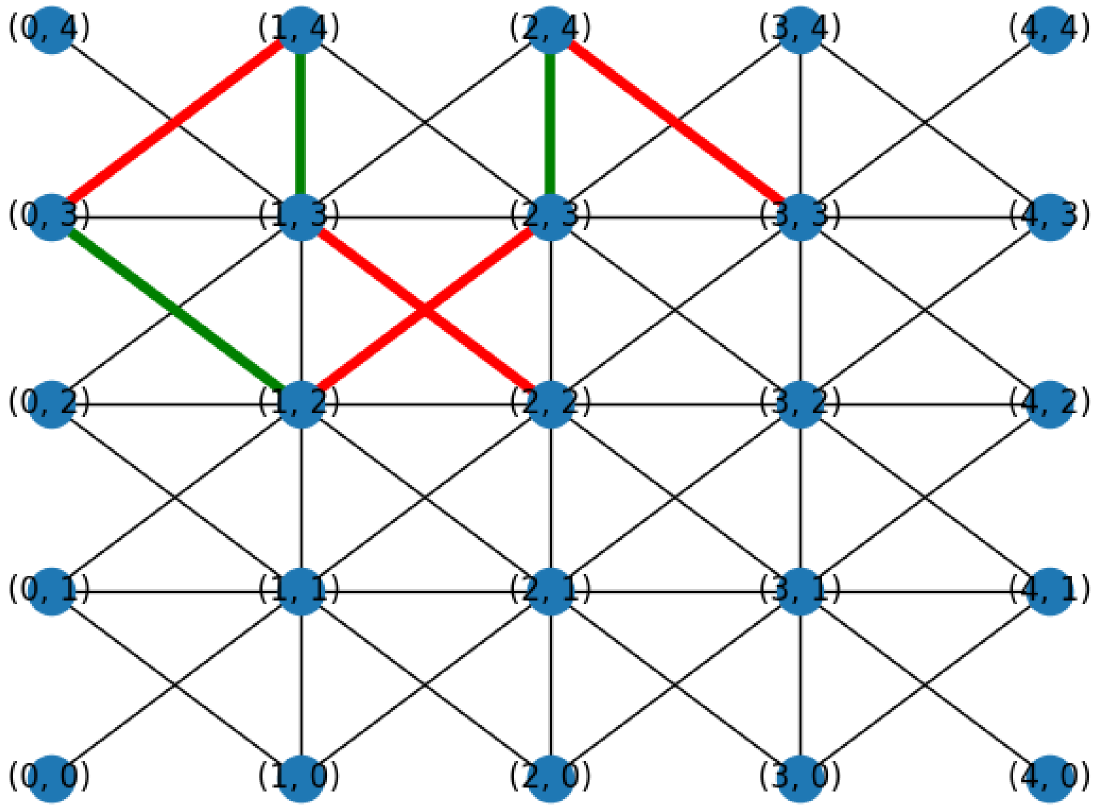

(0, 4)
(1, 4)
(2, 4)
(3, 4)
(4, 4)
(0, 3)
(1, 3)
(2, 3)
(3, 3)
(4, 3)
(0, 2)
(1, 2)
(2, 2)
(3, 2)
(4, 2)
(0, 1)
(1, 1)
(2, 1)
(3, 1)
(4, 1)
(0, 0)
(1, 0)
(2, 0)
(3, 0)
(4, 0)

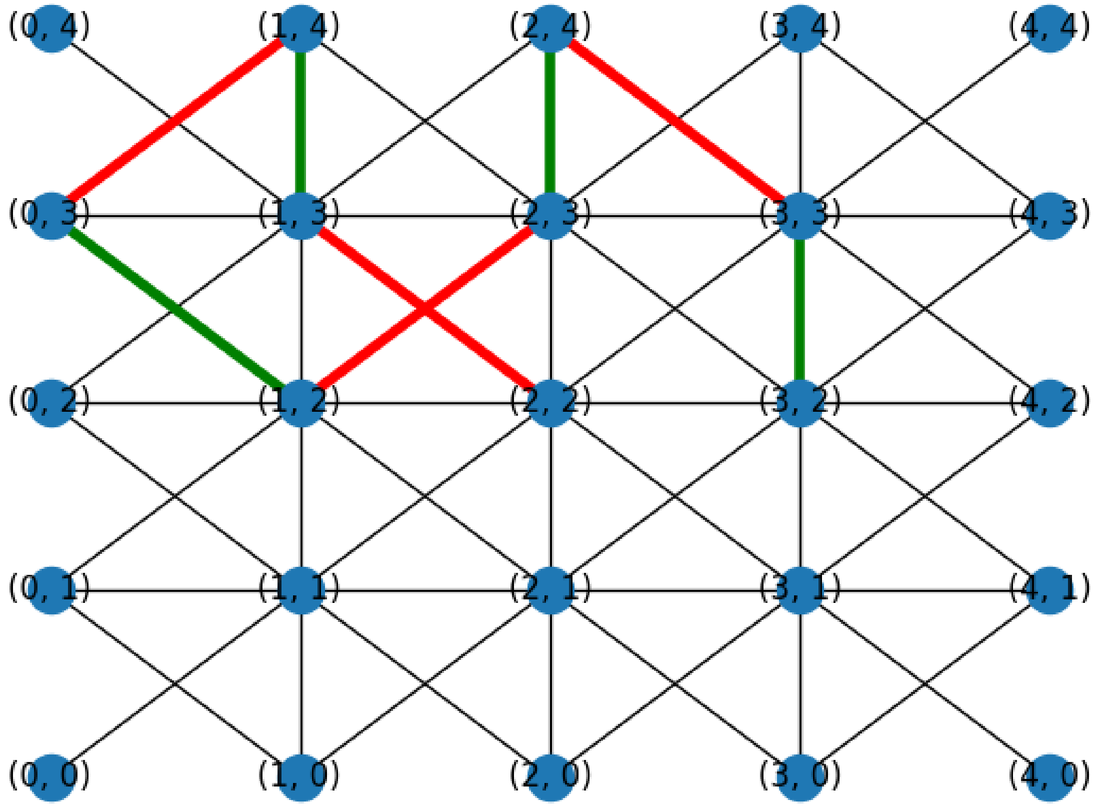

(0, 4)
(1, 4)
(2, 4)
(3, 4)
(4, 4)
(0, 3)
(1, 3)
(2, 3)
(3, 3)
(4, 3)
(0, 2)
(1, 2)
(2, 2)
(3, 2)
(4, 2)
(0, 1)
(1, 1)
(2, 1)
(3, 1)
(4, 1)
(0, 0)
(1, 0)
(2, 0)
(3, 0)
(4, 0)

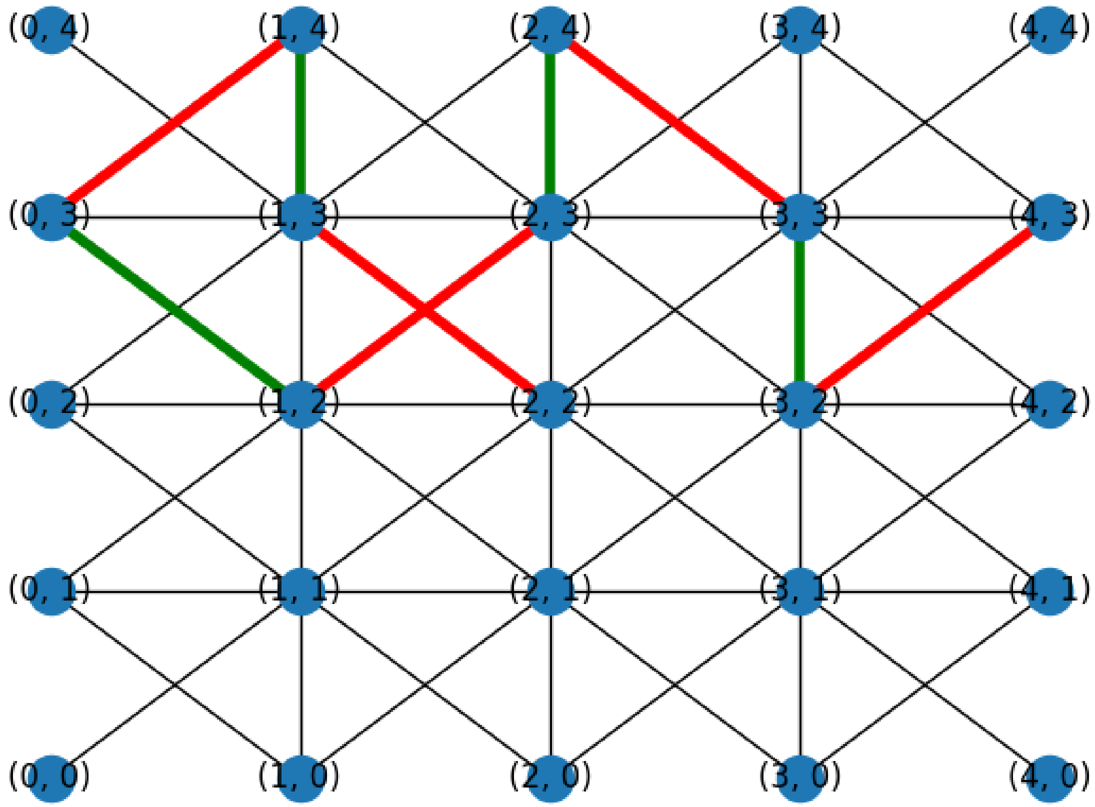

(0, 4)
(1, 4)
(2, 4)
(3, 4)
(4, 4)
(0, 3)
(1, 3)
(2, 3)
(3, 3)
(4, 3)
(0, 2)
(1, 2)
(2, 2)
(3, 2)
(4, 2)
(0, 1)
(1, 1)
(2, 1)
(3, 1)
(4, 1)
(0, 0)
(1, 0)
(2, 0)
(3, 0)
(4, 0)

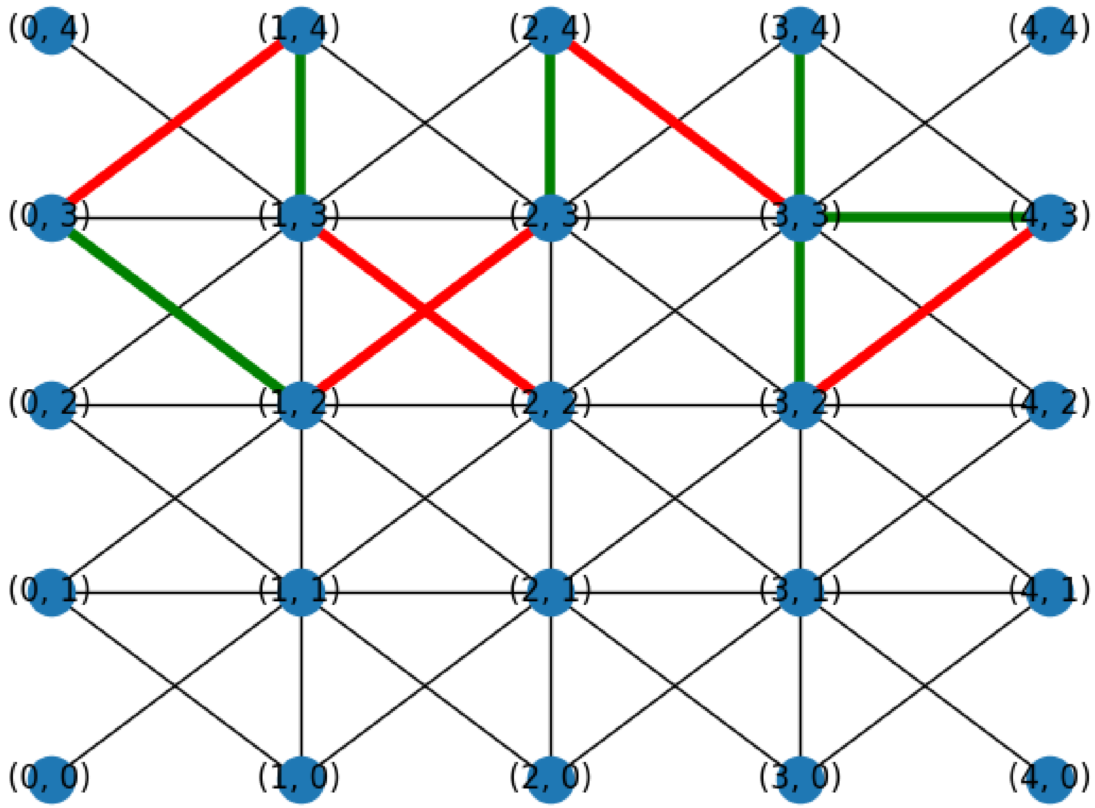

(0, 4)
(1, 4)
(2, 4)
(3, 4)
(4, 4)
(0, 3)
(1, 3)
(2, 3)
(3, 3)
(4, 3)
(0, 2)
(1, 2)
(2, 2)
(3, 2)
(4, 2)
(0, 1)
(1, 1)
(2, 1)
(3, 1)
(4, 1)
(0, 0)
(1, 0)
(2, 0)
(3, 0)
(4, 0)

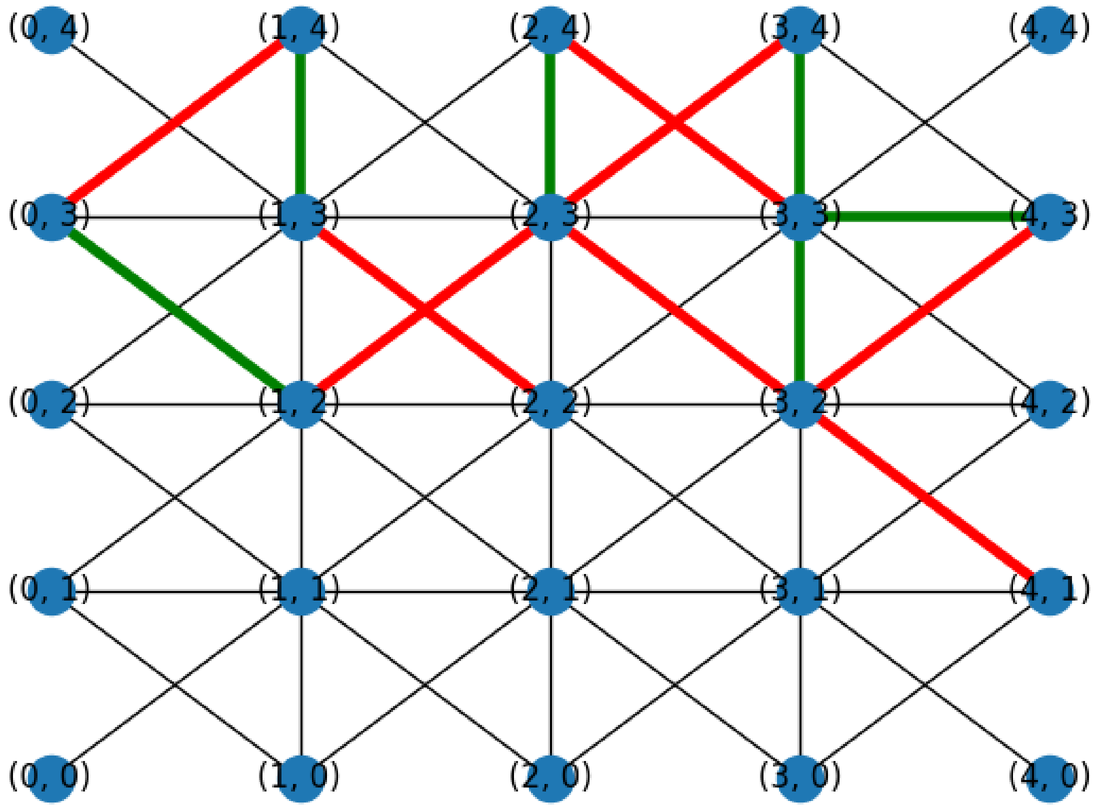

(0, 4)
(1, 4)
(2, 4)
(3, 4)
(4, 4)
(0, 3)
(1, 3)
(2, 3)
(3, 3)
(4, 3)
(0, 2)
(1, 2)
(2, 2)
(3, 2)
(4, 2)
(0, 1)
(1, 1)
(2, 1)
(3, 1)
(4, 1)
(0, 0)
(1, 0)
(2, 0)
(3, 0)
(4, 0)

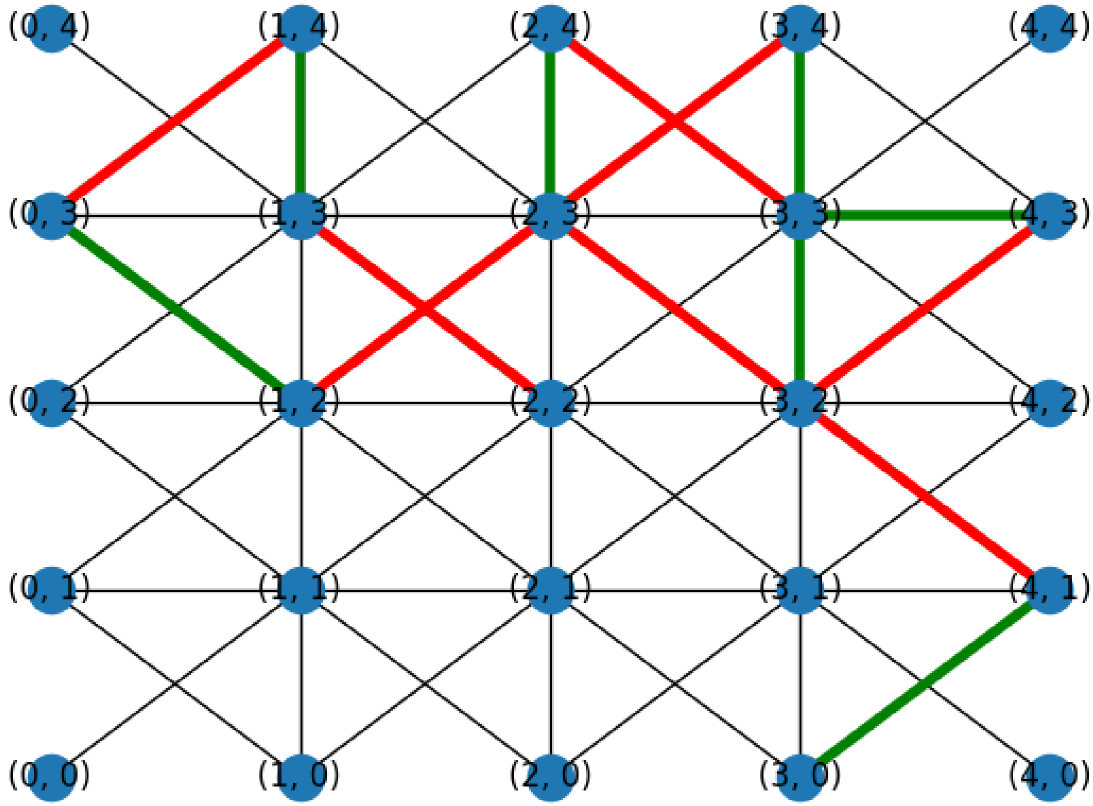

(0, 4)
(1, 4)
(2, 4)
(3, 4)
(4, 4)
(0, 3)
(1, 3)
(2, 3)
(3, 3)
(4, 3)
(0, 2)
(1, 2)
(2, 2)
(3, 2)
(4, 2)
(0, 1)
(1, 1)
(2, 1)
(3, 1)
(4, 1)
(0, 0)
(1, 0)
(2, 0)
(3, 0)
(4, 0)

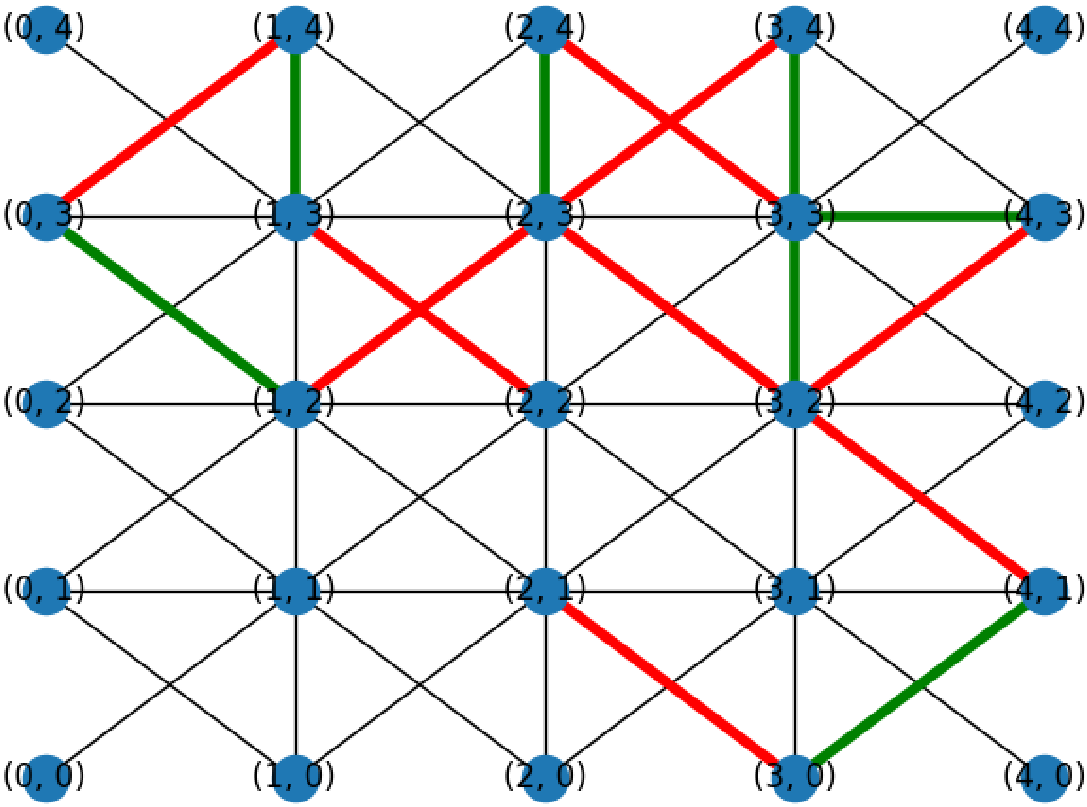

(0, 4)
(1, 4)
(2, 4)
(3, 4)
(4, 4)
(0, 3)
(1, 3)
(2, 3)
(3, 3)
(4, 3)
(0, 2)
(1, 2)
(2, 2)
(3, 2)
(4, 2)
(0, 1)
(1, 1)
(2, 1)
(3, 1)
(4, 1)
(0, 0)
(1, 0)
(2, 0)
(3, 0)
(4, 0)

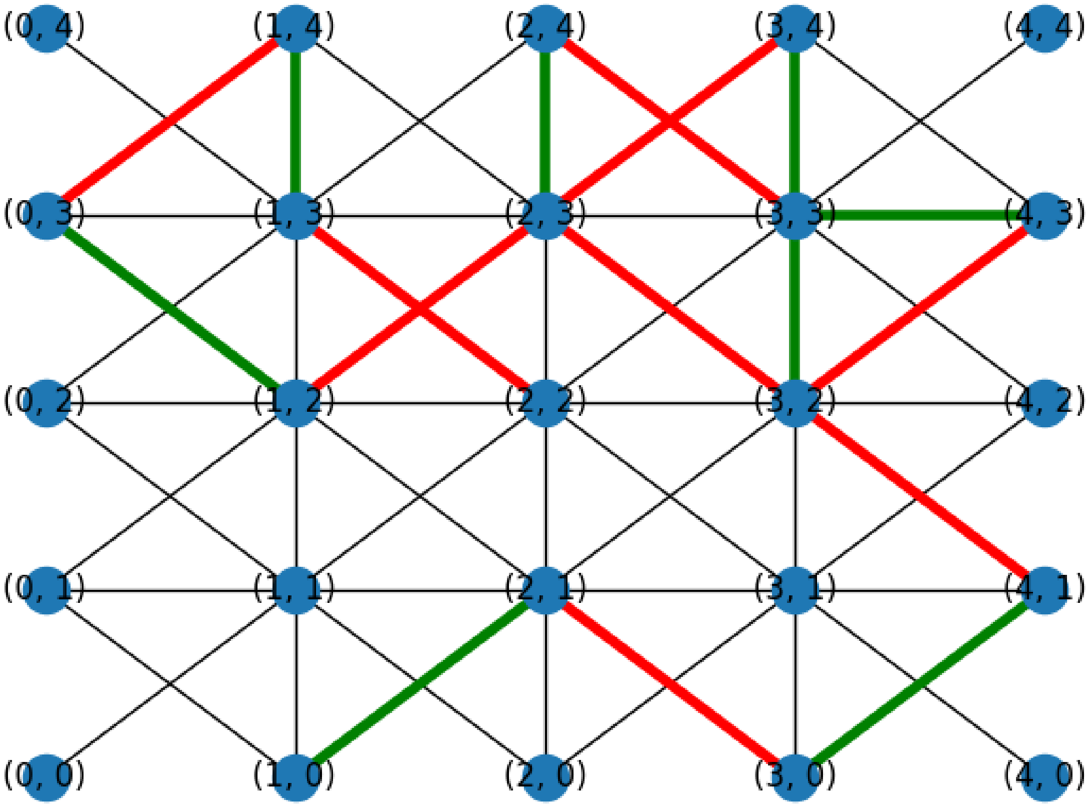

(0, 4)
(1, 4)
(2, 4)
(3, 4)
(4, 4)
(0, 3)
(1, 3)
(2, 3)
(3, 3)
(4, 3)
(0, 2)
(1, 2)
(2, 2)
(3, 2)
(4, 2)
(0, 1)
(1, 1)
(2, 1)
(3, 1)
(4, 1)
(0, 0)
(1, 0)
(2, 0)
(3, 0)
(4, 0)

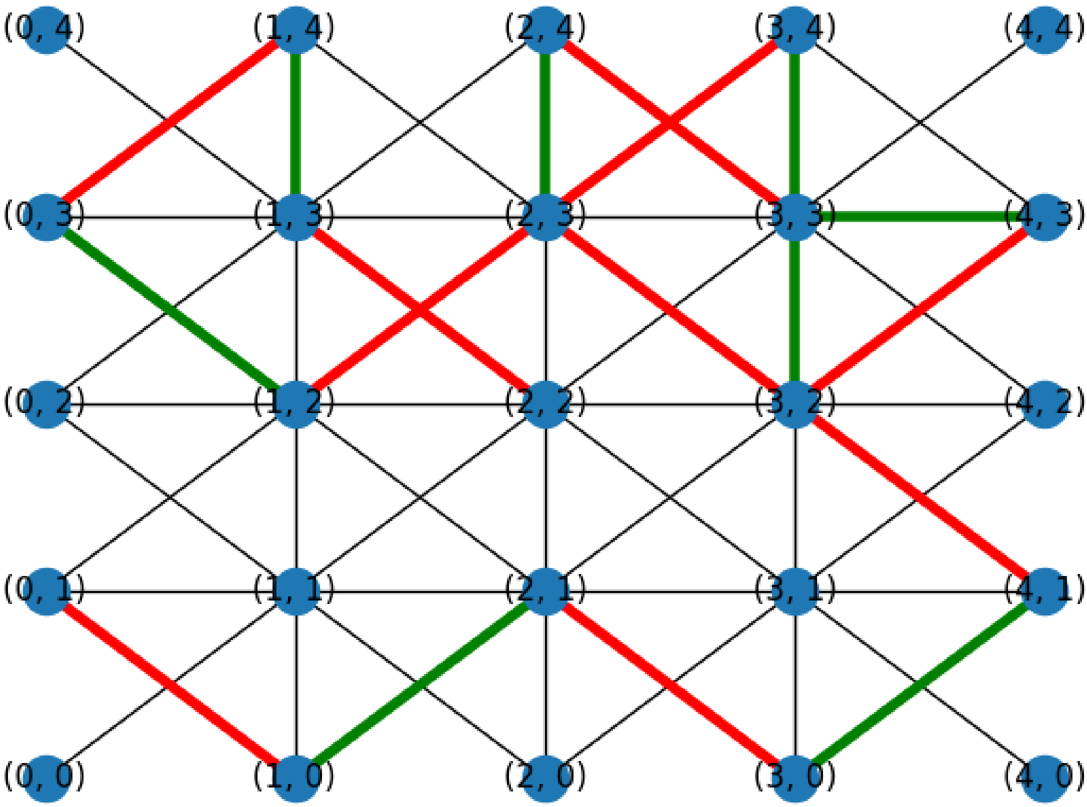

(0, 4)
(1, 4)
(2, 4)
(3, 4)
(4, 4)
(0, 3)
(1, 3)
(2, 3)
(3, 3)
(4, 3)
(0, 2)
(1, 2)
(2, 2)
(3, 2)
(4, 2)
(0, 1)
(1, 1)
(2, 1)
(3, 1)
(4, 1)
(0, 0)
(1, 0)
(2, 0)
(3, 0)
(4, 0)

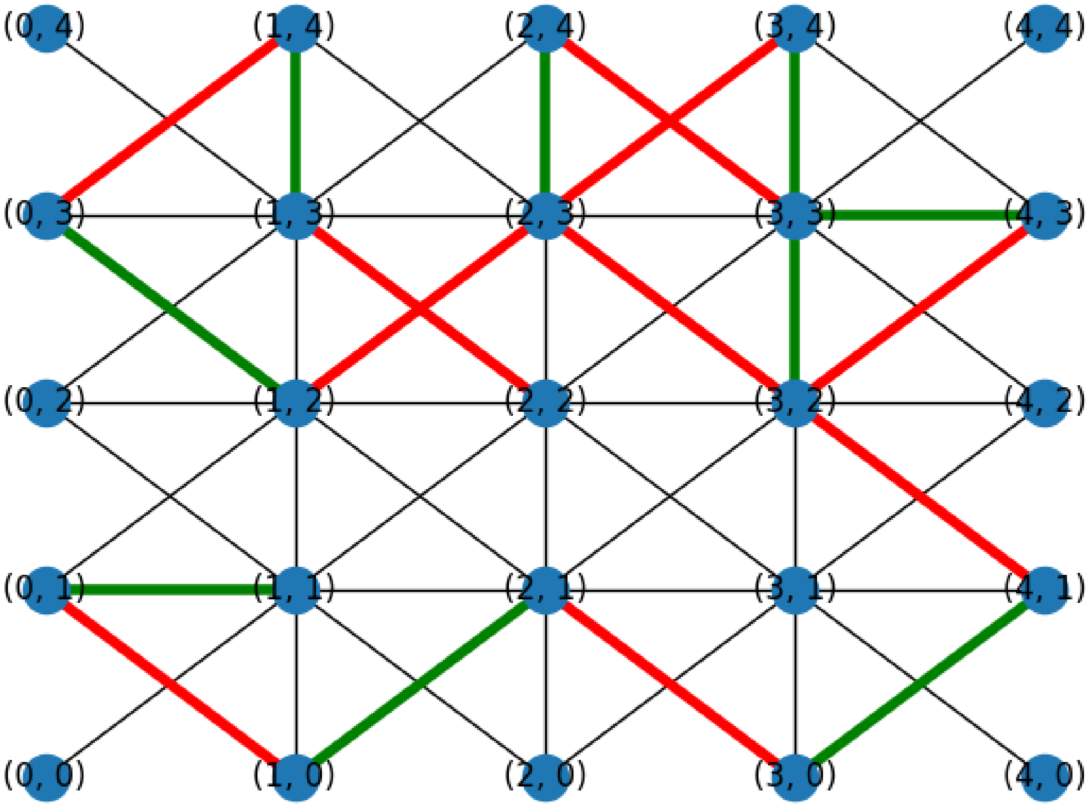

(0, 4)
(1, 4)
(2, 4)
(3, 4)
(4, 4)
(0, 3)
(1, 3)
(2, 3)
(3, 3)
(4, 3)
(0, 2)
(1, 2)
(2, 2)
(3, 2)
(4, 2)
(0, 1)
(1, 1)
(2, 1)
(3, 1)
(4, 1)
(0, 0)
(1, 0)
(2, 0)
(3, 0)
(4, 0)

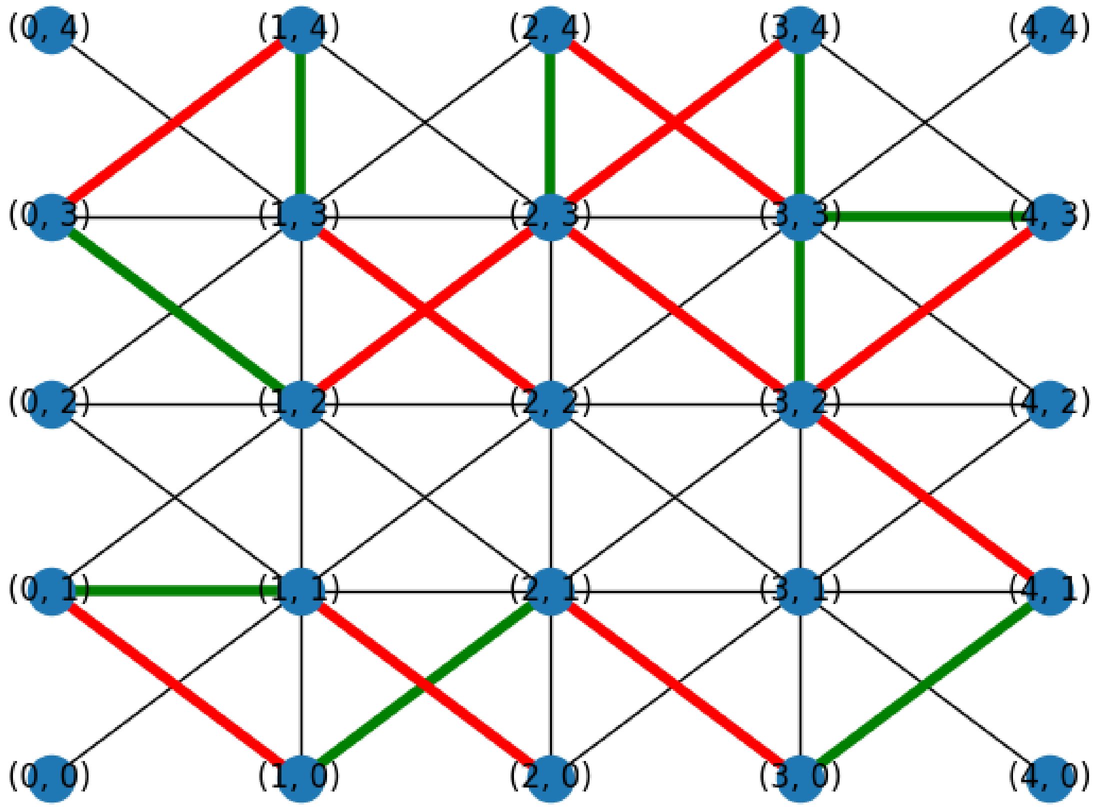

(0, 4)
(1, 4)
(2, 4)
(3, 4)
(4, 4)
(0, 3)
(1, 3)
(2, 3)
(3, 3)
(4, 3)
(0, 2)
(1, 2)
(2, 2)
(3, 2)
(4, 2)
(0, 1)
(1, 1)
(2, 1)
(3, 1)
(4, 1)
(0, 0)
(1, 0)
(2, 0)
(3, 0)
(4, 0)

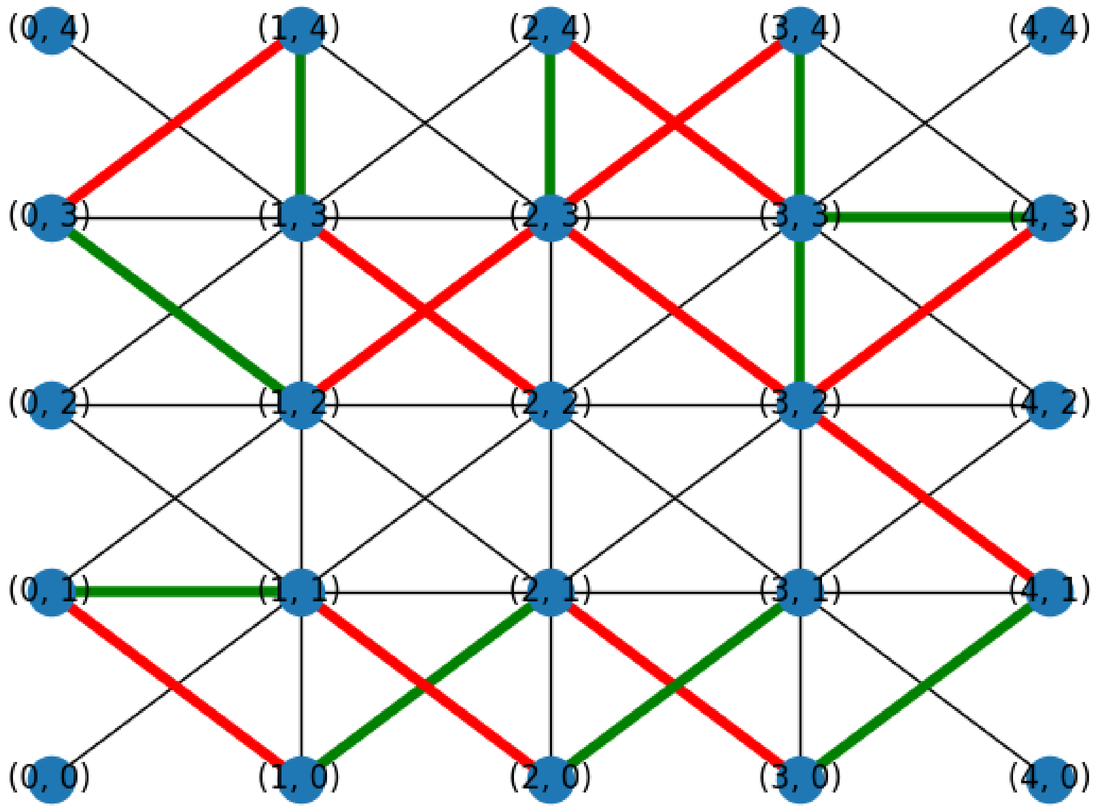

(0, 4)
(1, 4)
(2, 4)
(3, 4)
(4, 4)
(0, 3)
(1, 3)
(2, 3)
(3, 3)
(4, 3)
(0, 2)
(1, 2)
(2, 2)
(3, 2)
(4, 2)
(0, 1)
(1, 1)
(2, 1)
(3, 1)
(4, 1)
(0, 0)
(1, 0)
(2, 0)
(3, 0)
(4, 0)

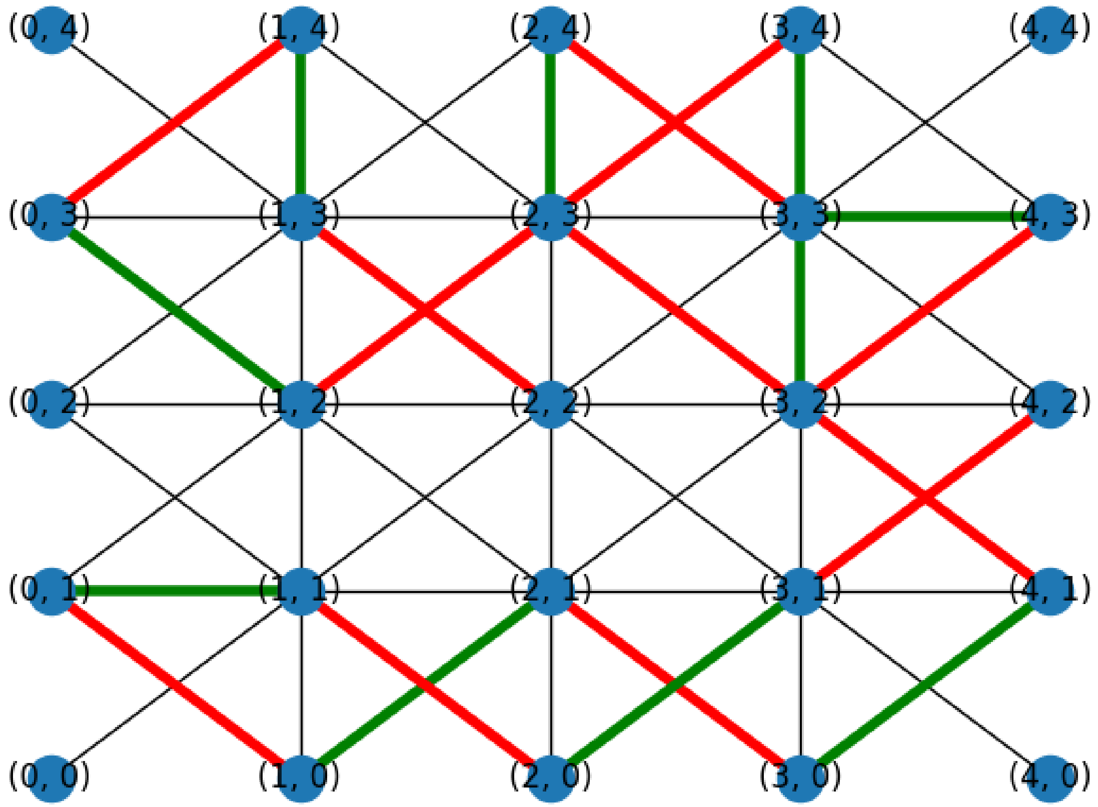

(0, 4)
(1, 4)
(2, 4)
(3, 4)
(4, 4)
(0, 3)
(1, 3)
(2, 3)
(3, 3)
(4, 3)
(0, 2)
(1, 2)
(2, 2)
(3, 2)
(4, 2)
(0, 1)
(1, 1)
(2, 1)
(3, 1)
(4, 1)
(0, 0)
(1, 0)
(2, 0)
(3, 0)
(4, 0)


(0, 4)
(1, 4)
(2, 4)
(3, 4)
(4, 4)
(0, 3)
(1, 3)
(2, 3)
(3, 3)
(4, 3)
(0, 2)
(1, 2)
(2, 2)
(3, 2)
(4, 2)
(0, 1)
(1, 1)
(2, 1)
(3, 1)
(4, 1)
(0, 0)
(1, 0)
(2, 0)
(3, 0)
(4, 0)


(0, 4)
(1, 4)
(2, 4)
(3, 4)
(4, 4)
(0, 3)
(1, 3)
(2, 3)
(3, 3)
(4, 3)
(0, 2)
(1, 2)
(2, 2)
(3, 2)
(4, 2)
(0, 1)
(1, 1)
(2, 1)
(3, 1)
(4, 1)
(0, 0)
(1, 0)
(2, 0)
(3, 0)
(4, 0)

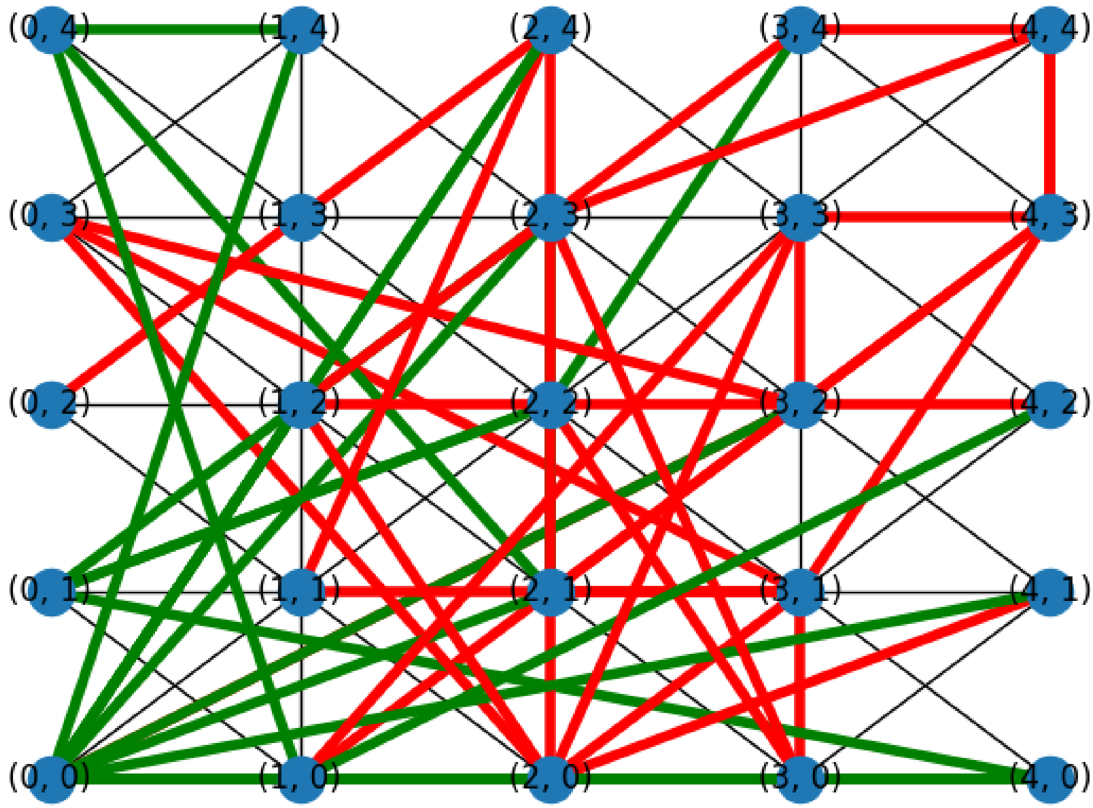

(0, 4)
(1, 4)
(2, 4)
(3, 4)
(4, 4)
(0, 3)
(1, 3)
(2, 3)
(3, 3)
(4, 3)
(0, 2)
(1, 2)
(2, 2)
(3, 2)
(4, 2)
(0, 1)
(1, 1)
(2, 1)
(3, 1)
(4, 1)
(0, 0)
(1, 0)
(2, 0)
(3, 0)
(4, 0)


(0, 4)
(1, 4)
(2, 4)
(3, 4)
(4, 4)
(0, 3)
(1, 3)
(2, 3)
(3, 3)
(4, 3)
(0, 2)
(1, 2)
(2, 2)
(3, 2)
(4, 2)
(0, 1)
(1, 1)
(2, 1)
(3, 1)
(4, 1)
(0, 0)
(1, 0)
(2, 0)
(3, 0)
(4, 0)



Metaheurystyka

Poniżej prezentuje się różnica pomiędzy podejściem dokładnym, sprawdzającym wszystkie możliwe opcje, a metaheurystyką, którą w moim przypadku jest metoda symulowanego wyżarzania. W przypadku metaheurystyki temperatura początkowa została ustawiona na 100, mnożnik zmiany temperatury w każdej iteracji na 0,95, maksymalna ilość iteracji na 100000. Rozwiązania sąsiadujące zostają wybierane jako zamiana losowej zmiennej decyzyjnej na losową wartość.



Niestety, jak możemy zauważyć rozwiązanie po 100 tysiącach iteracji nie jest nawet jakkolwiek zbliżone do rozwiązania uzyskanego przez solver CPLEX.

Wnioski

Metoda przeszukiwania całej dziedziny rozwiązań za każdym razem doprowadziła nas do wyniku spełniającego wszystkie ograniczenia, co w przypadku metaheurystyki się nie udało. Zaletą metaheurystyki jest jednak czas działania algorytmu.

Link do repozytorium z kodem: https://github.com/1mkmk/paper-soccer