20 listopada 2022
Autor: Robert Okuła
robert.okula@pg.edu.pl

1 Podstawy gry Go

Celem projektu jest stworzenie prostej implementacji gry Go w formie aplikacji knsolowej. Go jest abstrakcyjną strategiczną grą planszową dla dwóch graczy. Celem gry jest otoczenie („podbicie") więcej terytorium niż przeciwnik.
(Uproszczone) zasady gry:

1. Gracze grają na kwadratowej planszy (ja: *goban*), z 19 liniami przecinającymi planszę pionowo i 19 liniami przecinającymi ją poziomo (plansza 19x19 – 361 przecięć). Czasami rozmiar planszy to 13x13 albo nawet 9x9.
2. Gracze kładą białe i czarne kamienie na przecięciach planszy, jeden kamień na turę. Gracz, który gra przy użyciu kamieni czarnych zaczyna pierwszy.
3. Celem gry jest otoczenie największej powierzchni planszy wykorzystując własne kamienie.
4. Kamień, który został położony na planży pozostaje tam dopóki nie zostanie on wzięty do niewoli (zabity). Kamień jest zabity jeżeli jest on otoczony ze wszystkich czterech stron przez kamienie przeciwnika albo przez róg planszy. Wolne strony kamienia nazywane są *oddechami*, np. jeżeli kamień jest otoczony z dwóch stron przez kamień o przeciwnym kolorze to ma on dwa oddechy.
5. Gracz nie może położyć kamienia w sposób samobójczy (tj. kamień, który po położeniu od razu będzie pozbawiony oddechów). Gracz może położyć kamień na przecieciu bez oddechów tylko i wyłącznie w sytuacji gdy któryś z kamieni przeciwnika zostanie w ten sposób zabity.
6. Należy zachowywać zasadę Ko – *kamienie na planszy nie mogą powtarzać poprzedniego ułożenia kamieni. Ruchy, które wprowadzają planszę w ten sam stan są zabronione, należy wtedy wykonać w tej turze ruch w innej pozycji*.

Te zasady mogą być jednak dalej rozwinięte.

1.1 Oddechy

Jak wspomniano wcześniej, kamień jest zabity w momencie w którym jest on otoczony ze wszystkich czterech stron przez kamienie przeciwnika albo róg planszy (innymi słowy – jest on pozbawiony oddechów). Jednakże może wystąpić także inna sytuacja – kamienie ułożone pionowo lub poziomo (ale nie na skos) mogą tworzyć grupę. Grupa nie może zostać podzielona i kamienie w grupie współdzielą oddechy – można je zabić wyłącznie razem. Grupy mogą być rozwinięte nie tylko w jednej linii, ale także formować gałęzie – gracz może położyć dodatkowe kamienie na wszystkich przyleglych przecięciach, nie tylko na końcach.

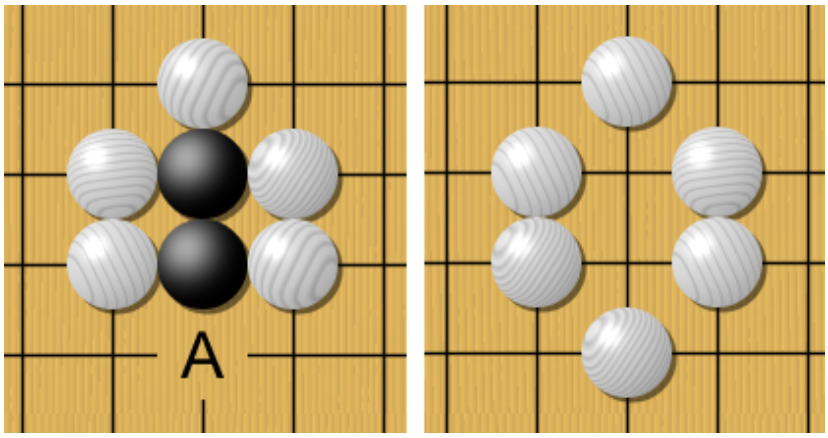

Figure 1: Proces zabicia krótkiej grupy kamieni. Źródło: Stevertigo, Wikicommons, CC BY-SA 2.0

1.2 Zasada Ko

Gracz nie może wykonać rchu który przywraca grę do poprzedniego układu kamieni na planszy – innymi słowy, nieskończona powtarzalna pętla z dwoma krokami nie jest możliwa. Taka sytuacja może się wydarzyć gdy występuje tzw. *walka Ko* – na planszy z lewej strony ilustracji czarny może zabić biały kamień zaznaczony czerwonym kółkiem. Jednakże, po tym ruchu to biały może w podobny sposób zabić kamień czarny, po wykonaniu tego ruchu plansza wraca do pierwotnego ustawienia. Taka sytuacja nazywa się Ko. Tak więc, zasada Ko wymaga, aby gdy jeden gracz wykorzysta sytuację Ko, drugi nie może wykonać tego samego od razu, musi poczekać do następnej rundy.

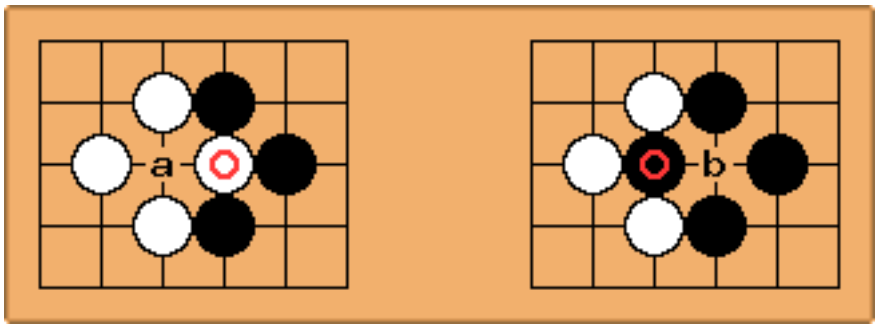

Figure 2: Walka Ko. Źródło: Sensei's Library

1.3 Liczenie punktów (oraz *Komi*)

W ogólności, dwa typy liczenia punktów są stosowane, jednakże w tym projekcie student powinien wykorzystać metodę Japońską. Gracze powinni zachować zabite kamienie, które nazywamy *więźniami*. Wynik stanowi liczba pustych przecięć otoczona przez kamienie gracza + liczba więźniów złapanych przez gracza. Gra jest zakończona gdy stan każdego kamienia jest ustalony w bezpośredniej rozgrywce albo gdy jeszcze trochę kamieni może zostać zabita, ale gracze mogą to uczynić w następnych rundach (i drugi gracz nie może się wybronić), więc można policzyć to tak jakby ruch już został wykonany.
Przykładowo: obraz nr 3 pokazuje uproszczoną grę pod koniec rozgrywki. A oraz B mogą być dodane bezpośrednio do wyniku każdego gracza, punkty z F nie należą do któregokolwiek z graczy. Grupa D kamieni białych została efektywnie zabita, gdyż czarny może położyć jedynie dwa kamienie i usunąć oddechy tej grupy. Grupa C (i dodatkowy kamień) zaś zmniejszyły teren podbity przez białego i tym samym jego wynik.

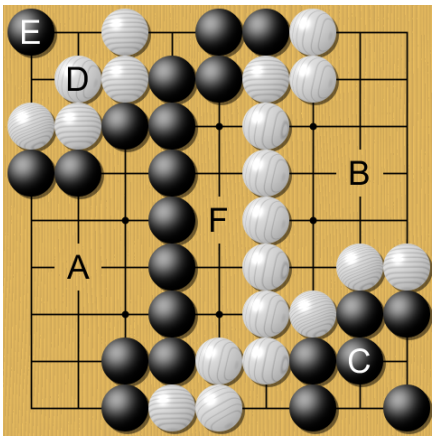

Figure 3: Uproszczona gra pod koniec rozgrywki. Źródlo: Scsc, Wikicommons, CC BY-SA 3.0

Ponadto, skoro czarny rusza jako pierwszy, czasami dodatkowe punkty są dane białemu aby zniwelować tę niesprawiedliwość. Taki dodatek nazywany jest *komi* i zwykle oznacza bonus 6.5 punkta. Czasami gdy gracze grają z handicapem (tj. gdy czarny rozpoczyna grę z kilkoma kamieniami położonymi już na planszy) bonus to 0.5 punkta, tylko po to aby rozbić ewentualny remis.

2 Ogólne wskazania dotyczące projektu

Program powinien być napisany z użyciem udostępnionego szablonu. Szablon umożliwia uzyskanie zaawansowanych możliwości w zakresie obsługi terminala w systemie Windows. Zabronione jest używanie instrukcji cin/cout/printf/scanf. W celu pisania na ekran i czytania z klawiatury, należy używać wyłącznie metod dostępnych w szablonie. Uwaga: udostępniony szablon działa w systemie Windows i nie ma możliwości łatwego przeniesienia go do innych systemów operacyjnych. Projekt należy zatem przygotować wyłącznie dla systemu Windows z wykorzystaniem szablonu.
Pamiętaj o przygotowaniu się do odbioru - koniecznie sprawdź wcześniej czy potrafisz sprawnie uruchomić swój program na komputerze w laboratorium!
Uwaga: projekt należy napisać tak, aby modyfikacja kodu (zazwyczaj poprzez zmianę wartości odpowiednich stałych) zmieniająca podstawowe parametry gry była łatwa. Przez parametry mamy na myśli na przykład (ale nie wyłącznie):

- zmiana położenia elementów interfejsu na ekranie (pozycja planszy),
- zmiana rozmiaru planszy (powinno być możliwe aby użyć mniejszej lub większej planszy).

2.1 Obsługa programu

Program powinien wykorzystywać klawiaturę w następujący sposób:

- `strzałki`: poruszanie kursorem na planszy;
- `q`: zamknięcie programu;
- `n`: rozpoczęcie nowej gry;
- `enter`: potwierdzenie wyboru i zakończenie tury gracza;
- `esc`: anulowanie obecnej akcji;
- `i`: położenie kamienia na planszy;
- `s`: zapis stanu gry;
- `l`: wczytanie stanu gry;
- `f`: zakończenie gry.

2.2 Wymagania obowiązkowe (5 punktów)

Wszystkie wymienione tutaj elementy należy zaimplementować. Brak któregokolwiek z poniższych elementów skutkuje otrzymaniem 0 pkt. z tego projektu.

(a) Wyświetlanie planszy (w jednym z następujących rozmiarów: 9x9, 13x13, 19x19) i legendy programu. Plansza powinna mieć obramowanie. Legenda powinna zawierać imię, nazwisko oraz numer albumu, listę zaimplementowanych funkcjonalności (w postaci wymienienia liter punktów z niniejszej instrukcji) oraz listy działających skrótów klawiszowych. Domyślnie legenda powinna być wyświetlona z lewej strony ekranu, natomiast plansza po prawej. Powinna być możliwość łatwego przesunięcia tych elementów poprzez zmianę odpowiednich stałych w kodzie programu.

(b) Poruszanie się po planszy z użyciem kursorów. W ramach legendy powinna być wyświetlona bieżąca pozycja kursora. Obsługa klawisza q kończącego program. Bieżąca pozycja kursora jest widoczna również na planszy; kursor nie może opuścić planszy (próby wyjścia poza planszę są ignorowane)

(c) Nowa gra. Wciśnięcie klawisza `n` powoduje rozpoczęćie nowej gry do stanu pierwotnego.

(d) Proste umieszczenie kamienia – wciśnięcie klawisza `i` powoduje położenie kamienia w pozycji kursora jeżeli komórka jest pusty i nie jest to oczywiste samobójstwo (kamień po położeniu ma co najmniej jeden oddech). Rozgrywka jest zawsze między dwoma osobami – po tym gdy czarny kamień został położony następuje tura białych itd.

(e) Proste zabicie – program powinien poprawnie wykryć zbicie pojedynczego kamienia, usunąć go z planszy i zwiększyć wynik odpowiedniego gracza..

2.3 Wymagania nieobowiązkowe (9 punktów)

(f) (**1 pt.**) *Zapis i przywrócenie stanu gry.* Wciśnięcie klawisza `s` pozwala użytkownikowi wpisać nazwę pliku do którego następuje zapis stanu gry. Naciśnięcie klawisza `l` powoduje zapytanie o nazwę pliku z którego następnie wczytuje stan gry (jeżeli istnieje). Ponadto należy zachować ułożenie kamieni: jeżeli gra jest zapisana przy np. trzech kamieniach na planszy, to gdy gra zostaje wczytana kamienie muszą znajdować się na dokładnie tych samych przecięciach.

(g) (**1 pt.**) *Zachowanie zasady Ko.* Gdy gracz spróbuje wykonać kolejny ruch w walce Ko w następnej rundzie po poprzednim graczu wykonującym także taki ruch, gra nie powinna pozwolić na zakończenie rundy. Jeżeli gra jest zapisana w trakcie walki Ko, to również powinna zachować stan tej walki i nie pozwolić temu graczowi na wykonanie ruchu w walce Ko po przeładowaniu.

(h) (**1 pt.**) *Zmiana rozmiaru planszy.* Gracz powinien móc wybrać każdy możliwy rozmiar planszy, dopóki mieści się on w ramach okna gry. Opcja ta powinna wpierw pokazać trzy standardowe, zdefiniowane opcje (9x9, 13x13 oraz 19x19), a następnie pokazać opcję czwartą po wyborze której będzie można podać własny rozmiar.

(i) (**1 pt.**) *Przesuwanie.* Gracz powinien mieć możliwość wyboru dowolnego rozmiaru planszy, nawet jeśli nie mieści się on w granicach okna programu. Ten punkt powinien być zaimplementowany po implementacji poprzedniego.

(j) (**2 pt.**) *Branie w niewolę.* Program powinien poprawnie wykryć zabicie każdej grupy kamieni, usunąć je z planszy i zwiększyć wynik odpowiedniego gracza.

(k) (**1 pt.**) *Edytor stanu gry.* Gracz powinien móc umieścić na planszy dowolny wstępny układ czarnych kamieni przed rozpoczęciem rozgrywki (gra z handicapem).

(l) (**2 pt.**) *Liczenie punktów.* Naciśnięcie klawisza `f` powinno zakończyć grę i pokazać wyniki punktowe, obliczone na podstawie zasad opisanych w Sekcji 1.3. Zasada *komi* również musi być wzięta pod uwagę – jeżeli gra rozpoczynana jest z handicapem, wartość dodatkowych punktów dla gracza grającego kamieniami bialymi musi być inna.

2.4 Wymagania dodatkowe (3 punkty)

Uwaga: poniższe wymagania są oceniane wyłącznie gdy zaimplementowane zostały wszystkie poprzednie punkty (a)-(l) **na łączną liczbę 14 punktów**.

(m) (**2 pt.**) *Wykrywanie zakończonej gry.* Program powinien wykryć czy jakiś inny ruch może być jeszcze wykonany bez obniżania punktacji jednego z dwóch graczy i przerwać grę, wyświetlając wyniki.

(n) (**1 pt.**) *Atari.* Program powinien oznaczyć kamienie z tylko jednym oddechem (przypomnienie: **grupy kamieni mają wspólne oddechy**), które mogą być zbite w następnym ruchu przeciwnika.

2.5 Uwagi końcowe

- Konfiguracja programu powinna umożliwiać łatwą zmianę wszelkich parametrów, nie tylko tych wyraźnie wskazanych w powyższym opisie. Przez łatwą zmianę rozumiemy modyfikację stałej w programie.
- Projekt może być napisany w sposób obiektowy, ale całkowicie zabronione jest używanie biblioteki standardowej C++ (w tym typu string, cin, cout, vector itp.) (Uwaga: typu string z biblioteki C++ nie należy mylić w biblioteką string.h z C – można używać funkcje znajdujące się w string.h).
- Obsługa plików powinna być zrealizowana przy użyciu biblioteki standardowej C (rodzina funkcji f???? - np. fopen, fread, fclose itd.) Nie można w tym celu używać mechanizmów C++ (np. fstream).
- Każdy fragment przedstawionego do oceny kodu powinien być napisany samodzielnie przez studenta. Nie jest dozwolone korzystanie z kodu znalezionego w Internecie, otrzymanego od innych osób lub napisanego z pomocą innych osób (z wyłączeniem pomocy uzyskanej na konsultacjach projektowych).
- Należy zwrócić uwagę na właściwy podział kodu na funkcje w celu unikania powielania kodu. Na przykład może się okazać, że w kilku wskazanych punktach przydatny będzie kod, który sprawdza czy zadana cyfra spełnia warunki wybranej reguły gry w zadanym polu planszy. W takim wypadku naturalne i wskazane jest umieszczenie takiego kodu wewnątrz funkcji wywoływanej następnie we właściwych miejscach.
- Stałe jednostki w programie powinny być opisane odpowiednimi komentarzami.