Programowanie obiektowe

Projekt grupowy pt.:

Bitwa o Pandorę

Skład grupy:

Jakub Smolarczyk

Wiktoria Wojdyło

Link do repozytorium:

https://github.com/maxster256/Battle_of_Pandora-updated

1. Krótki opis symulacji

Symulacja bitwy pomiędzy kolonizatorami (ludźmi), a plemieniem Na'vi. Po stronie ludzi występują jednostki takie jak zwykły żołnierz, robot i buldożer, a po stronie Na'vi łucznik oraz jeździec. Polem bitwy jest las, w którym występuje kilka rodzajów pól:

- Drzewo – nie może przez nie przejść żadna jednostka poza buldożerem
- Krzak – każda jednostka może przejść przez to pole, lecz ma ono wpływ na siłę jej ataku
- Pole puste – nie ma wpływu na jednostki i mogą one swobodnie się poruszać po tym rodzaju pola.

Symulacja trwa do momentu pokonania jednej ze stron lub do momentu upłynięcia podanej liczby iteracji. Symulacja została zaimplementowana w języku Java.

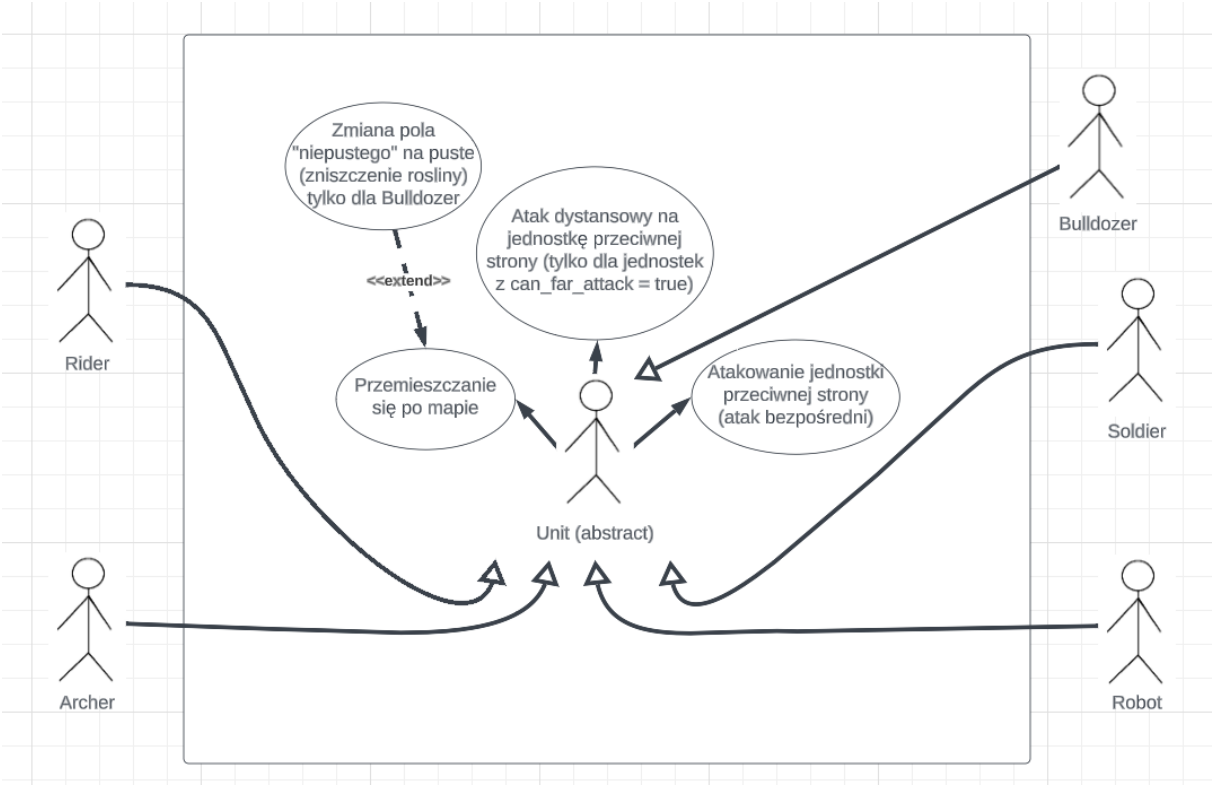
2. Dane wejściowe symulacji – Klasa Main

Symulacja posiada kilka opcji danych wejściowych mających wpływ na przebieg symulacji:

- Poziomy oraz pionowy rozmiar mapy
- Liczba łuczników, jeźdźców, żołnierzy oraz robotów
- Liczba iteracji symulacji
- Zagęszczenie mapy (stosunek liczby pól „niepustych" do liczby wszystkich pól wyrażony jako liczba od 1 do 100)

Ponadto w klasie main poza wprowadzeniem danych wejściowych tworzone są drużyny oraz wykonują się iteracje symulacji.

3. Implementacja interfejsu



Rys. 1 Diagram przypadków użycia symulacji „Bitwa o Pandorę”

Na powyższym rysunku 1 zaprezentowane są możliwe ruchy wszystkich jednostek. Klasy Rider, Archer, Robot, Soldier oraz Buldozer implementują pośrednio interfejs implementowany przez klasę abstrakcyjną Unit. Klasa buldożer poza poruszaniem się będzie w stanie również niszczyć roślinność na każdym polu na jakim się znajdzie, co przedstawia zależność <<extend>>. Ponadto w odróżnieniu od pozostałych jednostek jej atak będzie polegał na wyzerowaniu życia przeciwnika (natychmiastowym zabiciu go).

4. Dziedziczenie – Klasa Unit

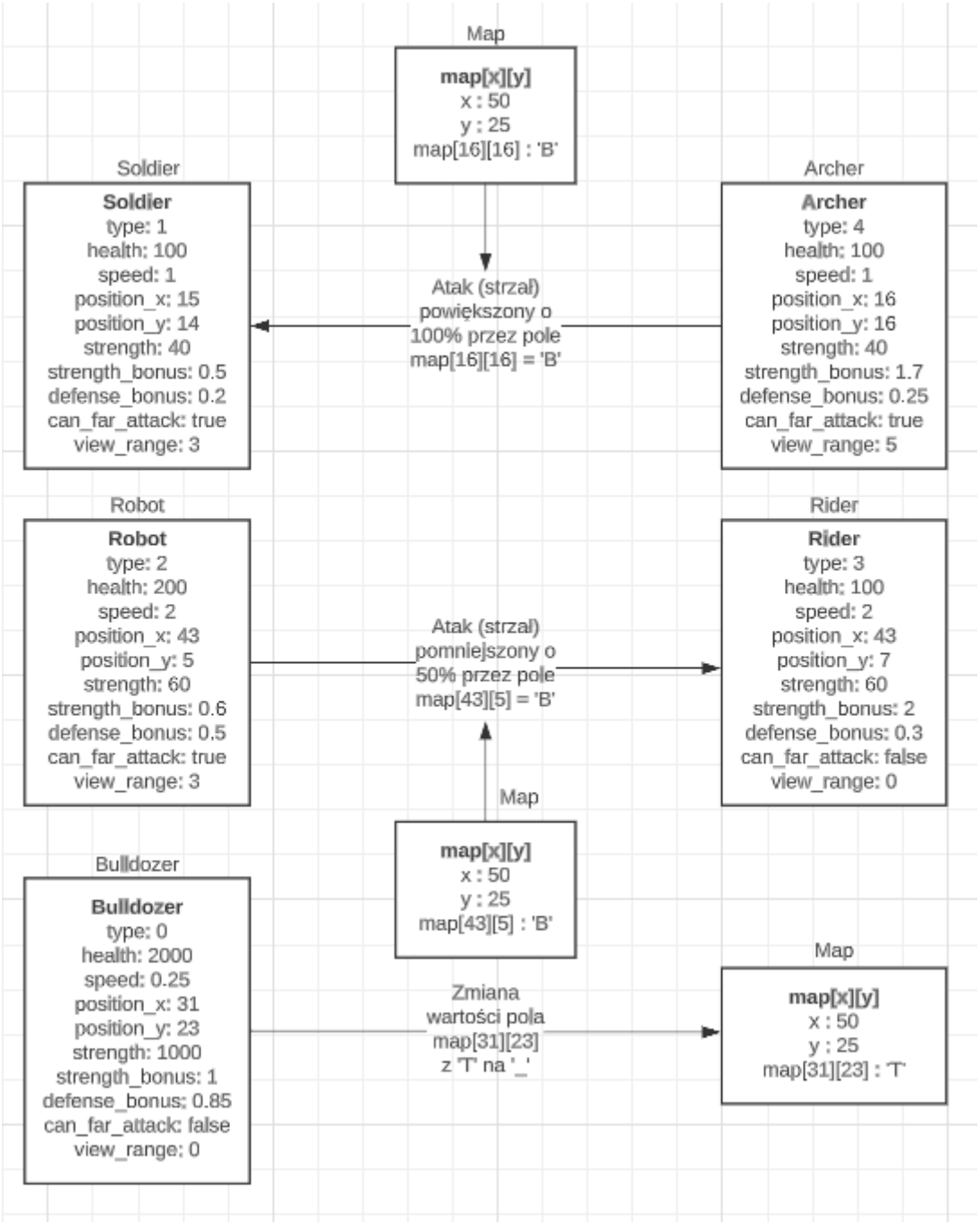
Klasy różnych typów jednostek czyli Bulldozer, Soldier, Robot, Archer, Rider dziedziczą po klasie abstrakcyjnej Unit następujące pola:

- Type – pole określające jakiego typu jest dana jednostka
- Health – prezentuje liczbę życia danej jednostki
- Speed – określa ile ruchów może jednostka wykonać w trakcie jednej iteracji symulacji np.: dla Speed = 2 może wykonać przemieszczenie po mapie 2 razy w trakcie jednej iteracji (w przypadku liczby niecałkowitej prawdopodobieństwo wykonania ostatniego możliwego ruchu wyraża się wzorem: Speed (mod 1) * 100%)
- Pos_x – pozycja pozioma na mapie danej jednostki

- Pos_y – pozycja pionowa na mapie danej jednostki
- Strength – określa ile obrażeń może dana jednostka zadać przeciwnikowi (ostateczna liczba zadanych obrażeń zależy od rodzaju ataku oraz pola, na którym stoi atakujący)
- Strength_bonus – mnożnik siły wpływający na jej ostateczną wartość
- Defense_bonus – stała wpływająca na prawdopodobieństwo zatrzymania ataku
- Can_far_attack – określa czy jednostka może wykonywać atak dystansowy
- View_range – określa na jaką odległość jednostka może wykonywać atak dystansowy

Ponadto w klasie Unit znajdują się metody dla wszystkich możliwych czynności jednostek takich jak przemieszczanie się, atak oraz atak dystansowy.

5. Diagram obiektów



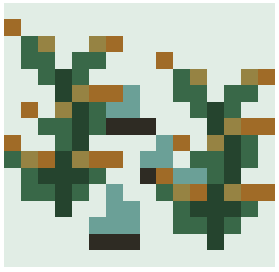
Rys. 2 Przykładowy diagram obiektów symulacji „Bitwa o Pandorę”

Na powyższym rysunku 2 przedstawione są przykładowe zachowania pomiędzy obiektami różnych klas. Pola mapy z krzakiem (czyli równe ‘B’) zmniejszają siłę ataku dla jednostek kolonizatorów, a dla jednostek Navi zwiększają. W przypadku prędkości mniejszej niż 1 dla buldożera prawdopodobieństwo wykonania przez niego ruchu będzie równe Speed*100%. Wartości pól mapy rozumiane są jako:

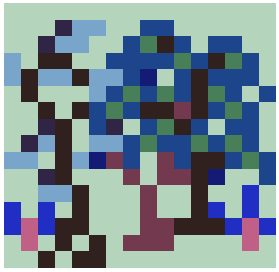
- B – krzak
- T – drzewo
- _ - puste pole

6. Klasa przeznaczona na mapę

W symulacji przewidziana jest także osobna klasa, w której wykonywane są m.in. metody generowania mapy, zmiany zawartości mapy (uwzględniająca działanie Buldożera) oraz wyświetlająca mapę. Wyświetlanie mapy odbywa się z pomocą m.in. JFrame oraz JPanel. Mapa składa się z prostokąta o jednolitym kolorze, na którym są umieszczone poniższe pliki .png o rozmiarze 16x16:



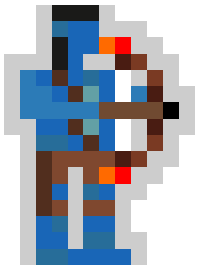
Rys. 3 plik .png krzaka



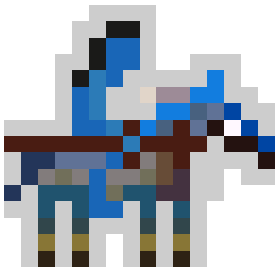
Rys. 4 plik .png drzewa

7. Wizualizacja jednostek w symulacji

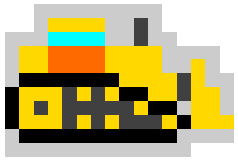
W wizualizacji symulacji poza zawartością mapy wyświetlane są także pliki .png poszczególnych typów jednostek:



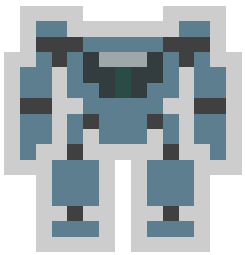
Rys. 5 plik .png lucznika



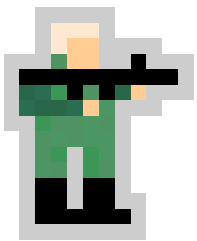
Rys. 6 plik .png jeźdźca



Rys. 7 plik .png buldożera



Rys. 8 plik .png robota



Rys. 9 plik .png żołnierza

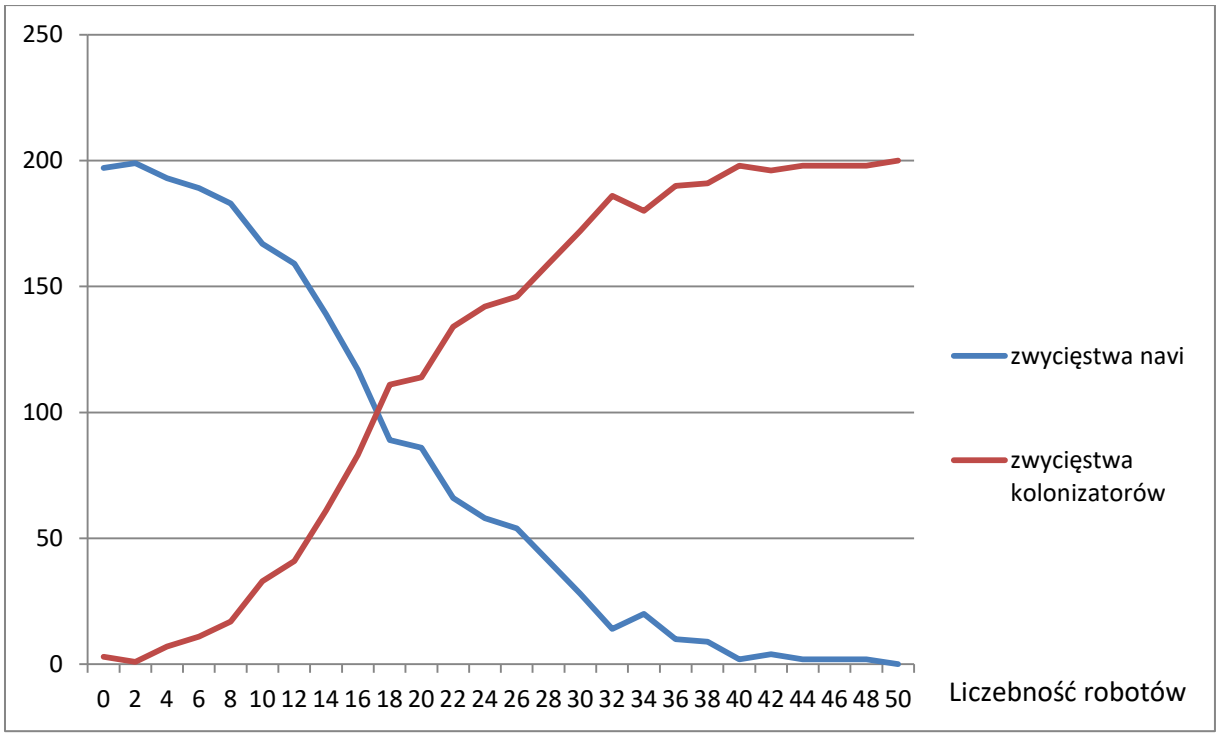
8. Badania symulacji

Na powstałej symulacji wykonaliśmy automatycznie za pomocą skryptów kilka badań, których celem było określenie zależności wyników symulacji od wybranego parametru.

W pierwszym badaniu sprawdziliśmy zależność wyników symulacji od liczby robotów na mapie. Dla pozostałych parametrów wynoszących odpowiednio:

- Density = 50
- Archer = Soldier = Rider = 25
- Rozmiar mapy: 50 x 25

Wykonaliśmy 200 powtórzeń symulacji dla liczby robotów równej 0, 2, 4 itd. aż do 50. Wyniki tego badania prezentuje poniższy wykres. Widać na nim, że największe zmiany nastąpiły pomiędzy wartością 10, a 26, dając mniej więcej równą liczbę wygranych dla obu drużyn w okolicy zaledwie 16 – 18 robotów, co wynika z ponadprzeciętnej szybkości i siły robotów w porównaniu z resztą jednostek.

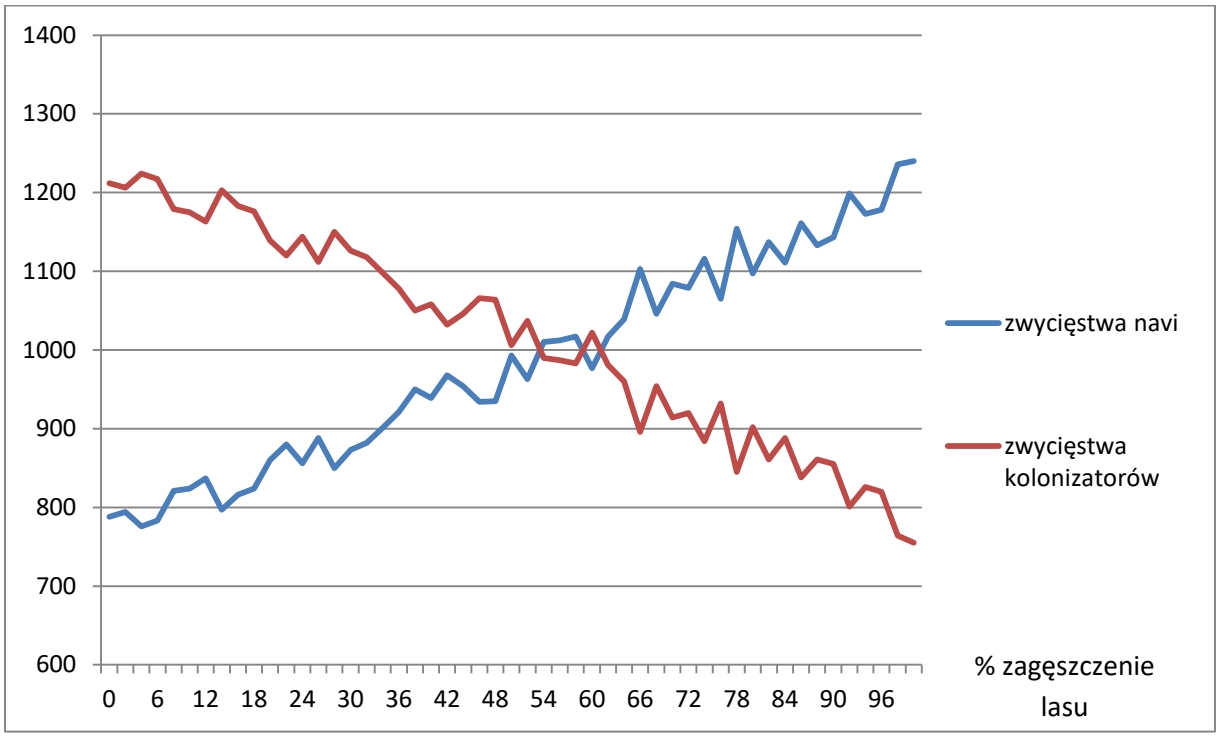


Rys. 10 wykres pierwszego badania

W drugim badaniu sprawdziliśmy zależność wyników symulacji od zagęszczenia lasu. Dla pozostałych parametrów wynoszących odpowiednio:

- Archer = 40
- Rider = 35
- Soldier = 34
- Robot = 28
- Rozmiar mapy: 50x25

Wykonaliśmy 2000 powtórzeń symulacji dla zagęszczenia 0, 2, 4 itd. aż do 100. Wyniki tego badania prezentuje poniższy wykres. Zauważyć na nim można liniową tendencję zależności liczby wygranych wśród obu drużyn od zagęszczenia lasu.

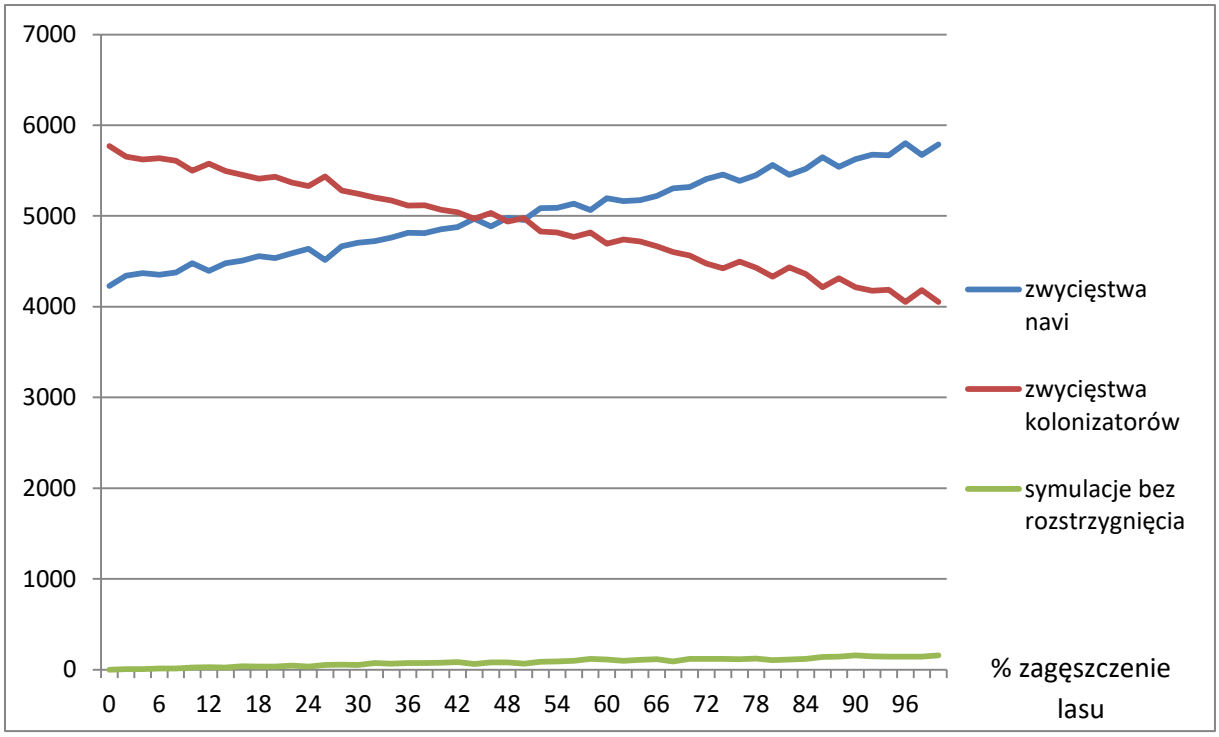


Rys. 11 wykres drugiego badania

Powtarzając badanie dla mniejszych wartości:

- Archer = 2
- Rider = 1
- Soldier = 2
- Robot = 1
- Rozmiar mapy: 20x10

Zauważyliśmy, że nawet dla długo działającej symulacji (ponad 100000 iteracji) pewna część z nich kończyła się bez rozstrzygnięcia, co przedstawia poniższy wykres (dla każdego zagęszczenia symulacja wykonana była 10000 razy). Wynika to prawdopodobnie z połączenia dużej liczby życia i bonusu obronnego buldożera oraz małej liczby atakujących go jednostek.

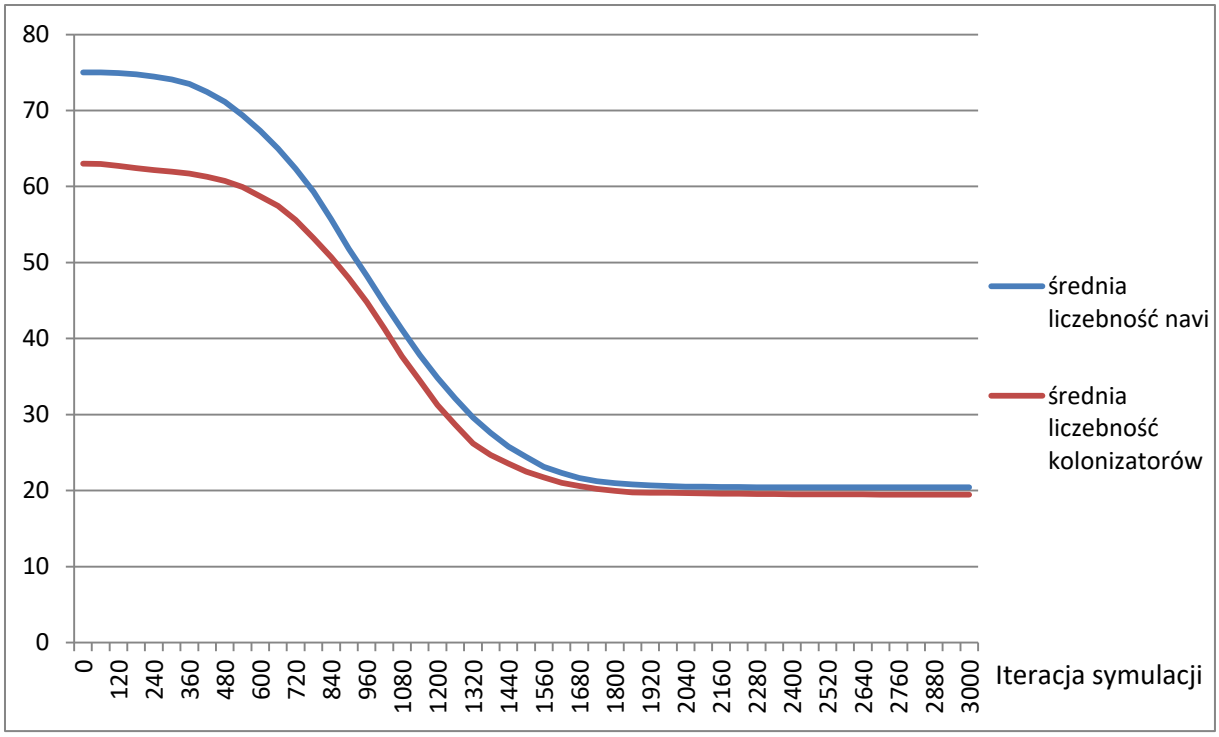


Rys. 12 wykres drugiego badania dla mniejszej mapy

W ostatnim badaniu sprawdziliśmy średnią liczebność drużyn podczas trwania symulacji. Dla parametrów wynoszących odpowiednio:

- Archer = 40
- Rider = 35
- Soldier = 34
- Robot = 28
- Rozmiar mapy: 50x25

Wykonaliśmy 200 powtórzeń symulacji trwającej 3000 iteracji. Średnią liczebność obu drużyn w trakcie trwania symulacji przedstawia poniższy wykres. Na jego podstawie można dojść do wniosku, że powyższe wartości parametrów dają szansę około 50% wygranej dla obu drużyn.



Rys. 13 wykres trzeciego badania