Symulacja sterowania windą

Karolina Matuszczyk, Adrian Kuśmierek

9 czerwca 2020

1 Wprowadzenie

Rozważany w niniejszym projekcie problem dotyczy procesu optymalizacji zarządzania wezwaniami i dyspozycjami grupy wind. W wysokich, wielokondygnacyjnych budynkach do wydajnego działania systemu wind konieczne jest logiczne powiązanie ich w jeden moduł transportowy. Symulując sterowanie zarówno grupą wind jak i pojedynczą windą warto wziąć pod uwagę dwa główne priorytety: skrócenie czasu oczekiwania pasażerów na przyjazd windy oraz dojazd na zadane piętro (oszczędność transportowa), a także oszczędność energetyczną. [4]

2 Opis problemu

W przypadku symulowania kontroli grupy wind głównym problemem jest przypisanie żądań pasażerów do określonych wind. Pasażer wciskając przycisk wywołania windy określa kierunek jazdy. System sterowania windą zazwyczaj zna wtedy jedynie kierunek zadany przez wywołanie pasażera, nie zna jednak ani liczby pasażerów ani miejsca docelowego przejazdu.[8]

Jedna z najpopularniejszych metod sterowania ruchem windy zakłada zatrzymywanie się na najbliższym wezwanym piętrze w kierunku jazdy windy. Stosowane są tutaj zbiorczość dół, kiedy winda zabiera pasażerów jadąc tylko w dół, zbiorczość góra, kiedy jedzie tylko w górę oraz zbiorczość góra-dół, kiedy winda zatrzymuje się na najbliższym piętrze w obie strony. Minimalizowana jest w ten sposób przebywana przez windę odległość. Jest to jednak stosunkowo mało optymalna metoda i może skutkować pojawianiem się kilku wind na jednym piętrze w tym samym momencie [8]. Dodatkowo bardzo dużo czasu poświęcane jest na przyspieszanie i hamowanie windy oraz otwieranie i zamykanie drzwi.

2.1 Najpopularniejsze algorytmy sterowania grupą wind

- Round-Robin (algorytm karuzelowy) - sekwencyjne przypisywanie pojawiających się wywołań kolejnym windom. Pierwsza winda otrzymuje pierwsze wywołanie, druga winda drugie itd. Algorytm zapewnia równomierne obciążenie każdej z wind. Możliwa jest modyfikacja algorytmu np. nadająca priorytet niektórym wywołaniom lub zakładająca, że winda bez wywołań powraca na parter.
- Zoning - polega na podziale budynku na strefy obsługiwane przez różne windy, każda z wind obsługuje jedną ze stref. Podział może być statyczny lub dynamiczny. Optymalność tej metody spada gdy jedna ze stref jest bardziej oblegana.

3 Przegląd rozwiązań w literaturze

Istnieje wiele możliwości symulowania systemu sterowania windą. Jednym z nich jest wykorzystanie silnika do tworzenia gier i symulacji jakim jest Unity3D. Jego zaletą jest fakt, że nie trzeba znać specjalistycznych narzędzi, a tylko język C# oraz podstawową obsługę API. Jest to dobry sposób na obrazowe przedstawienie symulacji za pomocą animacji czy akcji wykonywanych w czasie rzeczywistym przez ludzi. Silnik ten jest dobrym wyborem, jeżeli potrzebna jest symulacja ciągła i na przykład wykrywanie kolizji. Problemy zaczynają się pojawiać przy zwiększaniu prędkości symulacji. [7]

Lepszym wyborem byłyby płatne narzędzia SIMIO lub ARENA wykorzystywane właśnie do symulacji i wspierające wyświetlanie 3D. Za pomocą SIMIO udało się określić, że system sterowania windą będzie najbardziej wydajny, gdy dostosuje się go do otoczenia w którym jest wykorzystywany [5]. Wyniki tego badania mówią, że w symulacji lepiej skupić się na pojedynczym przypadku, czyli określonym budynku.

Istnieje wiele bardziej skomplikowanych rozwiązań. Zastosowanie czujników i liczników oczekujących na windę ludzi. Przekazywanie windzie informacji piętra docelowego przed wejściem do niej. Zastosowanie sztucznej inteligencji [6]. Rozwiązania te są jednak rzadko spotykane w życiu codziennym i nie będą brane pod uwagę w symulacji.

4 Potrzebne informacje

4.1 Rodzaje wind

W celu analizy sytuacji jak najbardziej zbliżonych do rzeczywistości dane o rodzajach wind zostały zaciągnięte z polskiego rynku wind. Prędkości wind oscylują od 0.63m/s dla dźwigów hydraulicznych [3] do 1.6m/s dla dźwigów elektrycznych [2]. Maksymalna liczba obsługiwanych pięter jest różnie podawana - od 8 do nawet 25 [2]. Symulacja skupi się jednak na budynkach do 8 pięter.

4.2 Sterowanie grupą wind

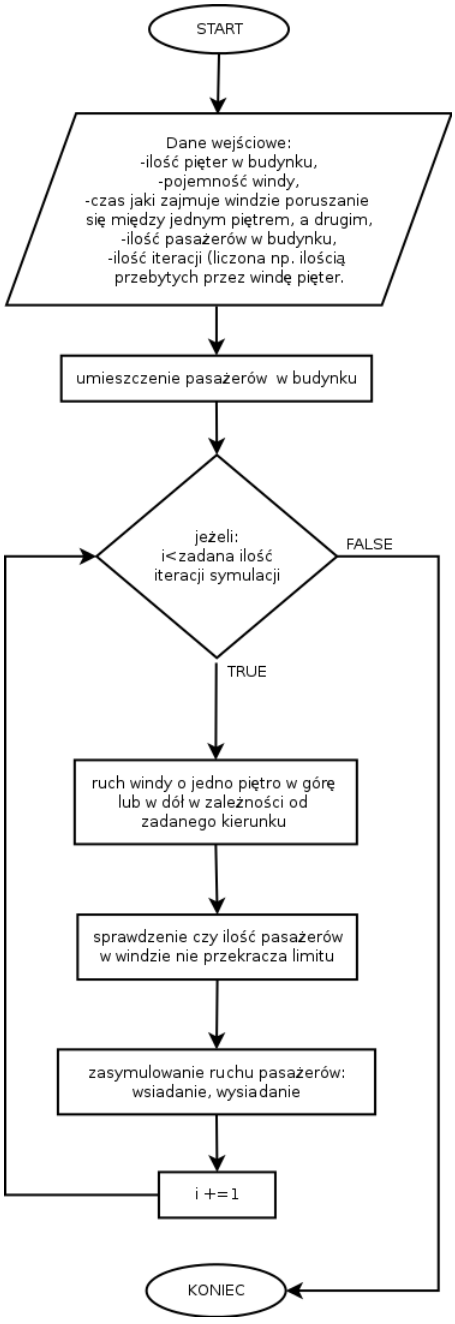
Najczęściej spotykanymi sposobami kontroli windy jest zbiorczość w dół, stosowana głównie w budynkach mieszkalnych i hotelach oraz zbiorczość góra/dół w biurowcach [3]. Symulowany przypadek ciężko przydzielić do jednej z tych kategorii, dlatego zostaną zbadane obie opcje w celu wybrania najlepszej. Do przydzielania żądań dla windy najczęściej stosowane jest przydzielenie żądania do najbliższej windy [3]. Nie jest to jednak optymalne i niektórzy producenci wprowadzają również algorytm Round-robin [1]. Tutaj również zostaną przetestowane obie opcje.

4.3 Założenia

W celu uproszczenia symulacji i ze względu na brak dostępu do dokładnych danych czas przyspieszania i zwalniania windy podczas ruchu oraz czas otwierania i zamykania drzwi został określony jako 5 sekund dla każdego przypadku windy.

5 Realizacja

5.1 Schemat działania

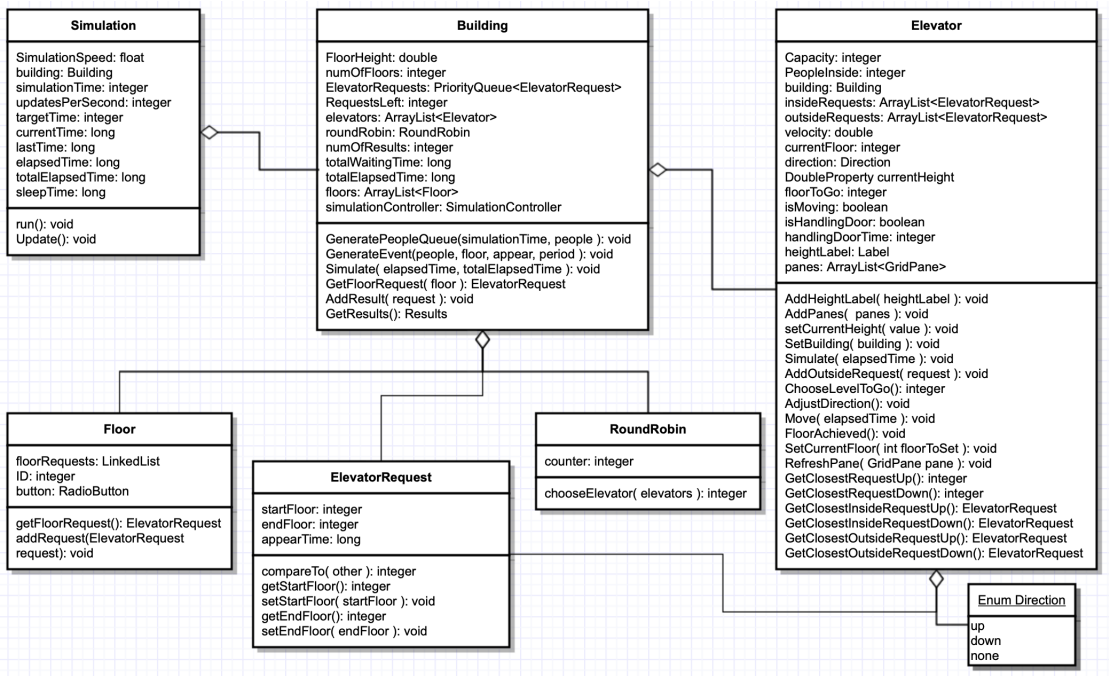


Rysunek 1: Schemat blokowy algorytmu

5.2 Implementacja kodu

Na początku symulacji program w określonym przedziale czasowym umieszcza określoną ilość pasażerów. W trakcie trwania symulacji pojawią się oni w przydzielonym wcześniej czasie i wyślą żądanie, aby winda przyjechała na ich piętro. W wersji podstawowej żądanie otrzymuje winda najbliższa, natomiast w wersji Round-Robin żądanie

zostaje przyporządkowane na zmianę do pierwszej i do drugiej windy. Winda nie ma informacji o ilości ludzi zarówno na zewnątrz jak i wewnątrz. Zna jedynie kolejność żądań. Wyszukiwane jest najbliższe żądanie wewnętrzne, a następnie czy po drodze znajduje się żądanie zewnętrzne. Winda zmierza do bliższego z tych dwóch. Po dotarciu na określone piętro pasażerowie opuszczają windę i wsiadają kolejni.

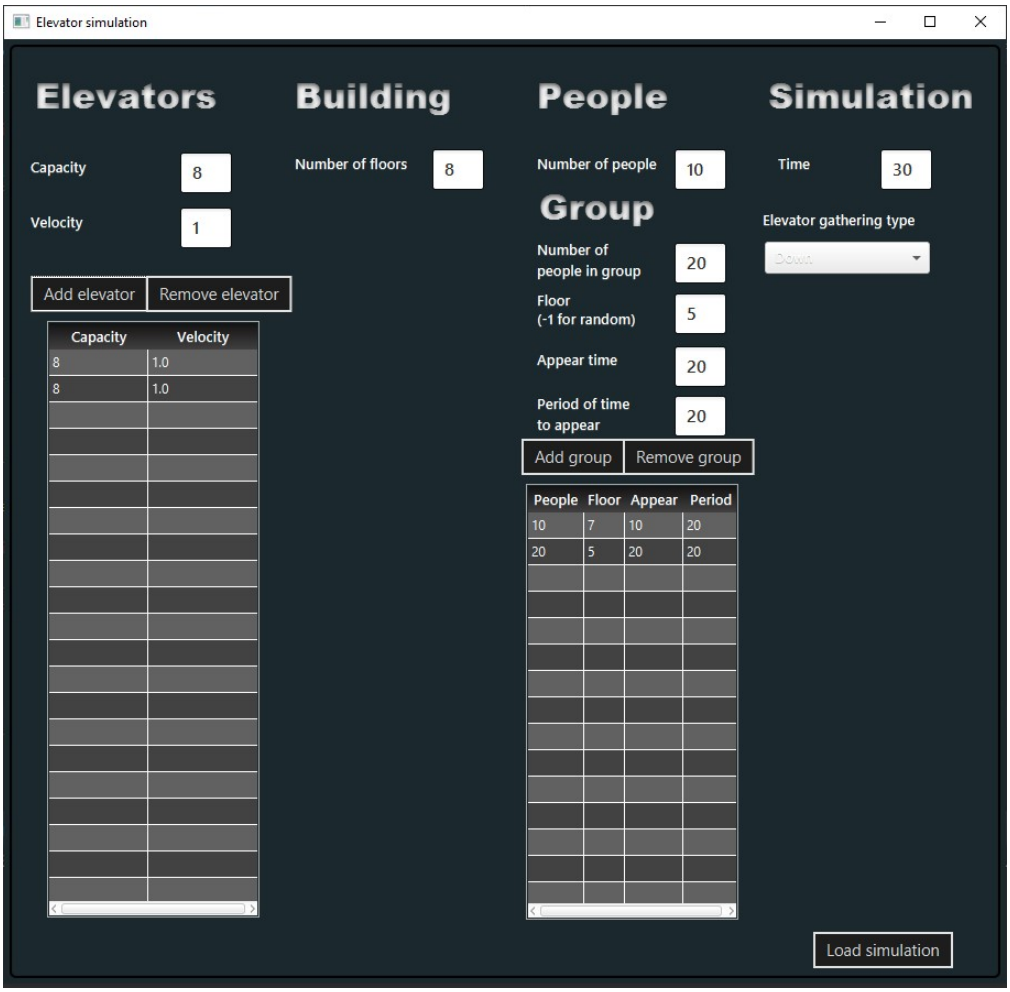


Rysunek 2: Diagram UML

5.3 Aplikacja

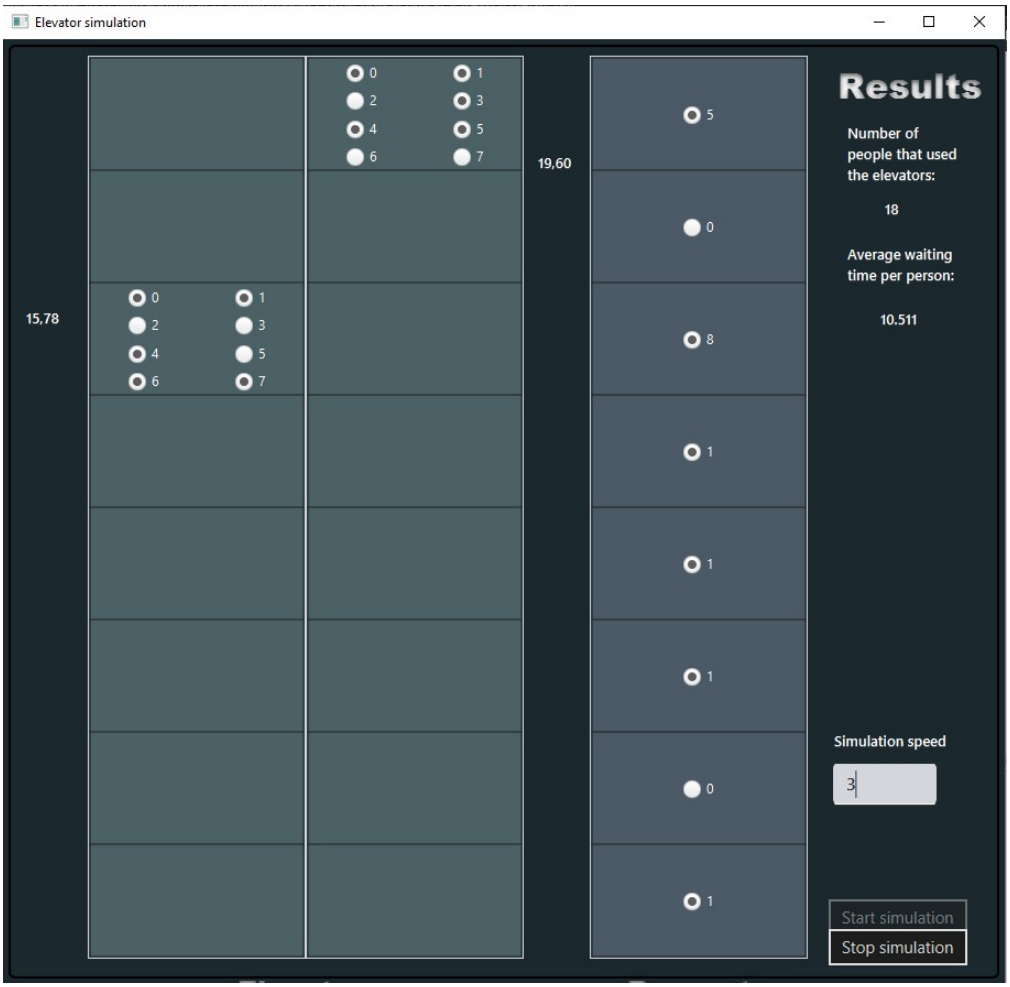
Aplikacja stworzona za pomocą języka Java pozwala użytkownikowi na wprowadzenie danych potrzebnych do symulacji:

- Windy - ilość, pojemność oraz prędkość
- Budynek - ilość pięter
- Pasażerowie - ilość przez czas trwania symulacji
- Grupy - do zasymulowania na przykład zakończenia wykładu na danym piętrze
- Symulacja - czas trwania oraz określenie zbiorczości wind



Rysunek 3: Okno aplikacji do wprowadzania danych

W trakcie trwania symulacji można zmieniać jej prędkość oraz na bieżąco odczytywać wyniki dla danej ilości pasażerów. Można również obserwować ruch pasażerów - ich żądania - oraz ruch wind - ich piętra docelowe.



Rysunek 4: Okno symulacji

5.4 Wyniki

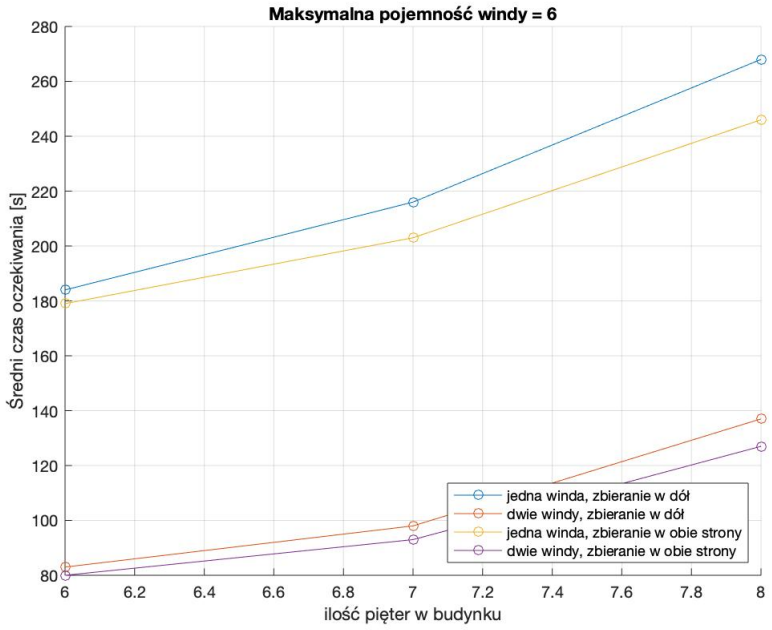
Przeprowadzono pomiary średniego czasu oczekiwania na windę dla zróżnicowanych wartości maksymalnej pojemności windy, liczby pięter w budynku oraz ilości wind. W pomiarach przyjęto prędkość windy wynoszącą 0.7 m/s oraz ilość osób równą 40 (w tym jedną 10 osobową grupę). Zmieniano również tryb pracy windy, która może zbierać pasażerów tylko przy ruchu w dół lub w obie strony. Każdy z wyników jest uśrednioną wartością z trzech pomiarów.

Wyniki pomiarów dla maksymalnej pojemności windy wynoszącej 6 osób, dla windy zbierającej pasażerów tylko w dół:

	liczba wind	
ilość pięter	**1**	**2**
6	184.077	83.135
7	216.937	98.668
8	268.127	137.998

Wyniki pomiarów dla maksymalnej pojemności windy wynoszącej 6 osób, dla windy zbierającej pasażerów przy ruchu w górę i w dół:

	liczba wind	
ilość pięter	**1**	**2**
6	179.946	80.684
7	203.868	93.844
8	246.250	127.872



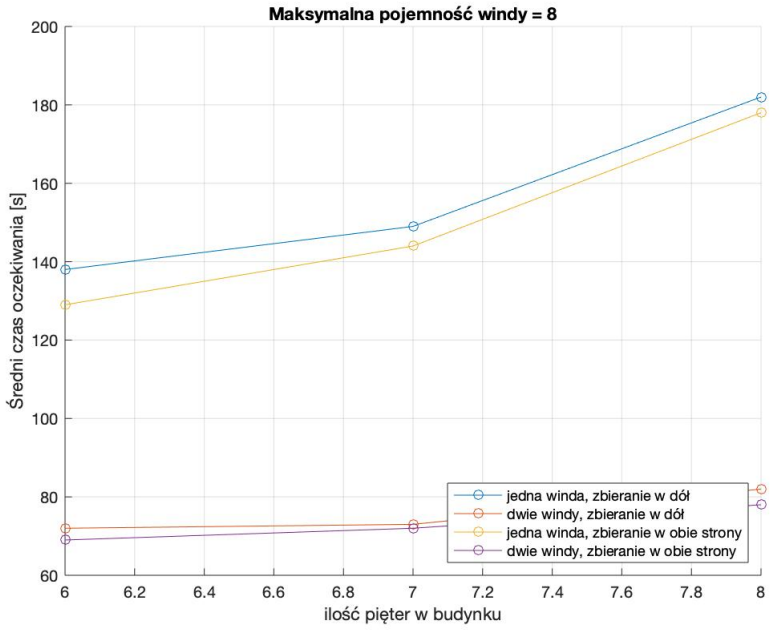
Rysunek 5: Wyniki pomiarów dla maksymalnej pojemności windy wynoszącej 6 osób

Wyniki pomiarów dla maksymalnej pojemności windy wynoszącej 8 osób, dla windy zbierającej pasażerów tylko w dół:

	liczba wind	
ilość pięter	**1**	**2**
6	138.205	55.447
7	149.065	73.529
8	182.775	82.564

Wyniki pomiarów dla maksymalnej pojemności windy wynoszącej 8 osób, dla windy zbierającej pasażerów przy ruchu w górę i w dół:

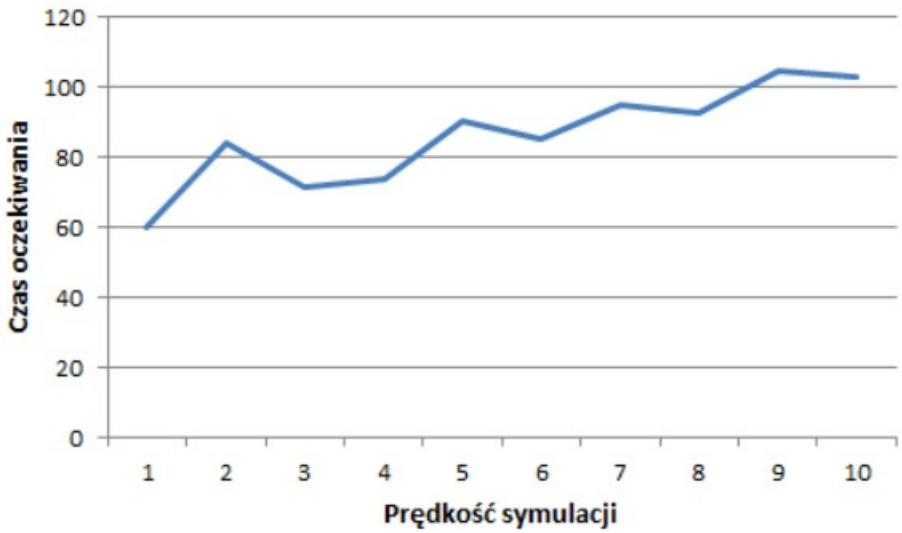
	liczba wind	
ilość pięter	**1**	**2**
6	129.554	69.791
7	144.135	72.665
8	178,868	77.865



Rysunek 6: Wyniki pomiarów dla maksymalnej pojemności windy wynoszącej 8 osób

5.5 Omówienie wyników

Podczas przeprowadzania symulacji napotkany został problem opisany już w symulacji przeprowadzanej za pomocą silnika Unity3D [7]. Ze względu na przedstawienie graficzne symulacji nie jest ona wyliczana idealnie, więc dokładniejsze wyniki otrzymuje się przy mniejszych prędkościach symulacji co pokazuje wykres 7. Zakładając wartość idealną wyliczoną dla prędkości symulacji 1, dla prędkości 10 widać prawie 70% wzrost dla tych samych danych



Rysunek 7: Różnica w czasach oczekiwania dla różnych prędkości symulacji

Same wyniki również są niedokładne. Symulacja przeprowadzana była na losowych danych. Za każdym razem pasażerowie wsiadali i wysiadali na innych piętrach. Rozwiązaniem byłoby przeprowadzenie większej ilości prób, dokładne określenie pasażerów lub dłuższy czas samej symulacji dla większej ilości osób.

6 Wnioski

Analiza uzyskanych wyników wykazała że stosując algorytm Round Robin przydzielania zapytań do wind dla większej ilości wind (dwóch wind) oraz trybu zbierania pasażerów w górę i w dół średni czas oczekiwania na windę jest mniejszy względem trybu zbierania pasażerów tylko w dół. Różnice te nie są jednak znaczne. Może to być spowodowane wyborem pięter docelowych przez pasażerów. Zbiorczość w dół stosowana jest w budynkach mieszkalnych, gdzie rzadko występuje ruch między piętrami, a częściej na piętro 0. Większe znaczenie w takim przypadku ma z całą pewnością dodanie drugiej windy. Bardzo często zmniejszało to czas oczekiwania o połowę.

Literatura

[1] https://automatykab2b.pl/temat-miesiaca/43652-rewolucja-w-technice-windowej/strona/1.

[2] https://www.schindler.com/pl/internet/pl/mobilne-rozwiazania/produkty/windy.html.

[3] http://www.gmv.pl/info-dzwig.html.

[4] Ł. Furgała, K. Kolano, and V. Mosorov. Model dynamicznego sterowania winda z wykorzystaniem serwera centralnego. *Informatics Control Measurement in Economy and Environment Protection*, 7:107–112, 2017.

[5] Marcelo Henriques, António Vieira, Luís Dias, Guilherme Pereira, and José Oliveira. Analysis of an elevator system using discrete event simulation: Case study. *International Journal for Quality Research*, 13:823–836, 11 2019.

[6] Jana Koehler and Daniel Ottiger. An ai-based approach to destination control in elevators. *AI Magazine*, 23, 10 2002.

[7] Jiří Polcar, Petr Horejsi, Pavel Kopeček, and Muhammad Latif. *Using Unity3D as an Elevator Simulation Tool*, pages 0517–0522. 01 2017.

[8] Srikumar Ramalingam, Arvind U. Raghunathan, and Daniel Nikovski. Submodular function maximization for group elevator scheduling. *ArXiv*, abs/1707.00617, 2017.