|  |  |  |
| --- | --- | --- |
|  | Politechnika Krakowska im. Tadeusza Kościuszki  Wydział Fizyki, Matematyki i Informatyki |  |
| **Pracownia Problemowa** | | |
| **Projekt:** Aplikacja harmoniująca przy równoczesnej optymalizacji harmonogramu dla egzaminów studentów. | | |
|  | Studia niestacjonarne  Informatyka  2014/2015 | Wykonał: Michał Zaborowski,  Łukasz Lech  Nr albumu: 88563,111389 |

**Spis treści**

Spis treści

[2 Opis problemu 3](#_Toc419142481)

[3 Podstawowe zagadnienia, wykorzystana technologia, algorytm, opis innych rozwiązań 4](#_Toc419142482)

[3.1 Wykorzystana technologia 4](#_Toc419142483)

[3.2 Opis użytego algorytmu 4](#_Toc419142484)

[3.3 Opis innych rozwiązań 4](#_Toc419142485)

[4 Graficzny interfejs użytkownika 6](#_Toc419142486)

[5 Diagramy funkcjonowania aplikacji 7](#_Toc419142487)

[6 Wyniki przeprowadzonych testów 7](#_Toc419142488)

[6.1 Koszt algorytmu 7](#_Toc419142489)

[7 Testy jednostkowe 9](#_Toc419142490)

[8 Bibliografia 9](#_Toc419142491)

# Opis problemu

Projekt dotyczy rozwiązania problemu układania planu zajęć (timetabling). Należy rozłożyć wszystkie egzaminy w taki sposób aby każdy ze studentów mógł podjąć egzaminy należące do niego. Zakładamy, iż harmonogram trzeba rozłożyć maksymalnie w 14 przedziałach czasowych (time slot). Projekt zakłada ograniczenia twarde (które muszą być spełnione). Do ograniczeń twardych należy:

Żaden uczeń nie może mieć dwóch egzaminów w tym samym terminie

Ranking harmonogramu będzie oceniany na podstawie najlepszego rozłożenia egzaminów w przedziałach czasowych tak aby przerwa dla każdego studenta między egzaminami była jak największa. Koszt oceniany jest na podstawie wag:

| Symbol | Waga | Opis |
| --- | --- | --- |
| w0 | 16 | Egzaminy sąsiadujące |
| w1 | 8 | Egzaminy oddalone o jeden przedział czasowy |
| w2 | 4 | Egzaminy oddalone o dwa przedziały czasowe |
| w3 | 2 | Egzaminy oddalone o trzy przedziały czasowe |
| w4 | 1 | Egzaminy oddalone o cztery przedziały czasowe |

Do rozwiązania projektu, wykorzystany zostanie plik sta-f-83-stu.txt, który zawiera dane ze szkoły St Andrew's Junior High School.

# Podstawowe zagadnienia, wykorzystana technologia, algorytm, opis innych rozwiązań

## Wykorzystana technologia

Aplikacja napisana została na urządzenia przenośne firmy Apple z systemem iOS używając Objective-C jako języka programowania. Do napisania programu został wykorzystany program XCode firmy Apple. XCode jest to pakiet dla dewelopera zapewniający narzędzia do tworzenia aplikacji na platformę iOS oraz Mac OS X. XCode jest ściśle zintegrowany z frameworkami Cocoa oraz Cocoa Touch. Pakiet XCode zawiera również narzędzie Instruments do analizy wydajności i analizy aplikacji. Początkowo dane studentów oraz egzaminów miały być przechowywane w bazie danych, iOS posiada wysokopoziomowy framework CoreData, który odpowiada za komunikację z SQLite. Niestety po testach wydajnościowych oraz wykorzystaniu wielowątkowości zrezygnowałem z tego podejścia, zastąpione zostało ono serializacją w formacie binarnym, przy użyciu biblioteki FastCoding[[1]](#footnote-2), dzięki temu otrzymałem dużo lepsze rezultaty oraz brak konfliktów.

## Opis użytego algorytmu

Spośród dostępnych metodologii służących do rozwiązywania problemu Timetable Scheduling Problem, zdecydowałem się na wykorzystanie algorytmu Simulated Annealing ze względu na niezłe wyniki[[2]](#footnote-3), łatwość implementacji, w większości przypadków daje dobre rozwiązanie. Algorytm polega na symulowania wyżarzania, losowo zakłócane są zmienne niezależne i wybierana jest najlepsza wartość funkcji dla przypadkowego zbioru zmiennych. Po wielu próbach zbiór, który daje najlepsze wartości funkcji staje się centrum, wokół którego wykonuje się ruchy dla następnej wartości temperatury (odchylenie standardowe generatora liczb losowych). Temperatura jest redukowana i przeprowadza się nowe próby. Bar- dzo ważnym zagadnieniem dla algorytmu wyżarzania jest wybór temperatury początkowej, końcowej a także sposób obniżania temperatury.

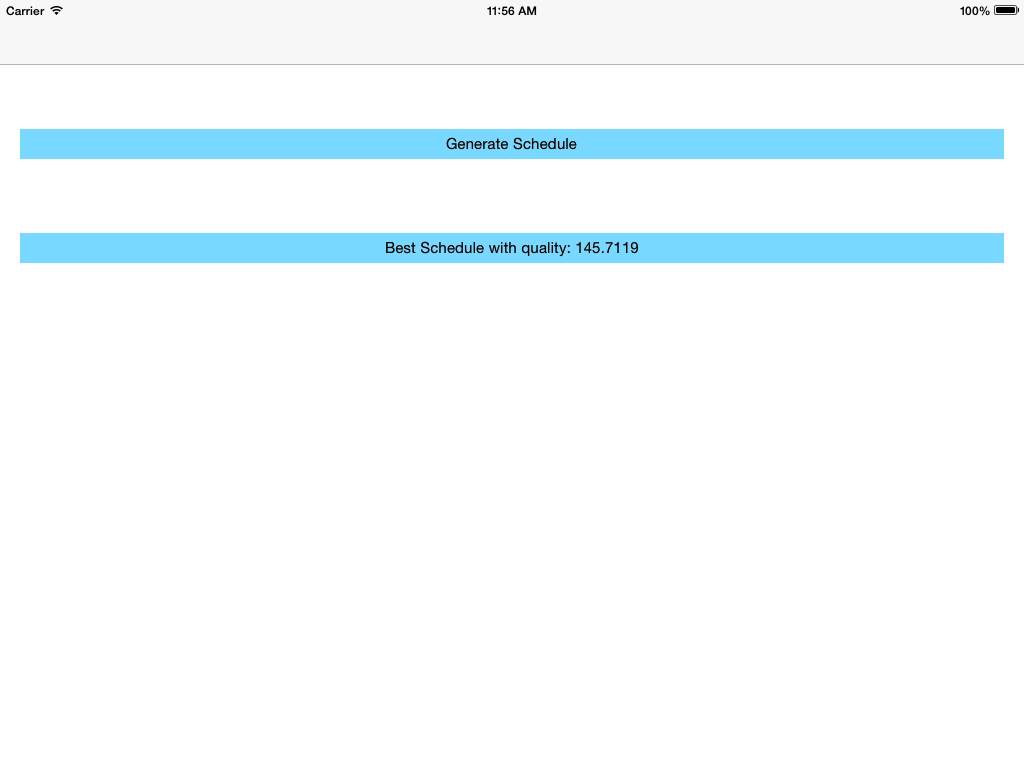
Gdy temperatura jest duża to każde rozwiązanie jest akceptowane - błądzenie losowe, natomiast gdy temperatura jest bliska zeru to akceptowane są tylko roz- wiązania lepsze. Algorytm nie jest podatny na utknięcie w lokalnym minimum funkcji gdyż dopuszcza zaakceptowanie rozwiązań gorszych od aktualnego.

## Opis innych rozwiązań

Problem stworzenia harmonogramu jest powszechnie znany jako NP-trudny. Trudność rozwiązywania problemów NP-zupełnych polega na tym, że liczba obliczeń, jakie trzeba wykonać, by znaleźć rozwiązanie, gwałtownie rośnie wraz z rozmiarem problemu. Poza podejściem opisanym w powyższym punkcie wybranym przeze mnie, istnieje kilka innych sposobów rozwiązania tego problemu, przykładowe z nich:

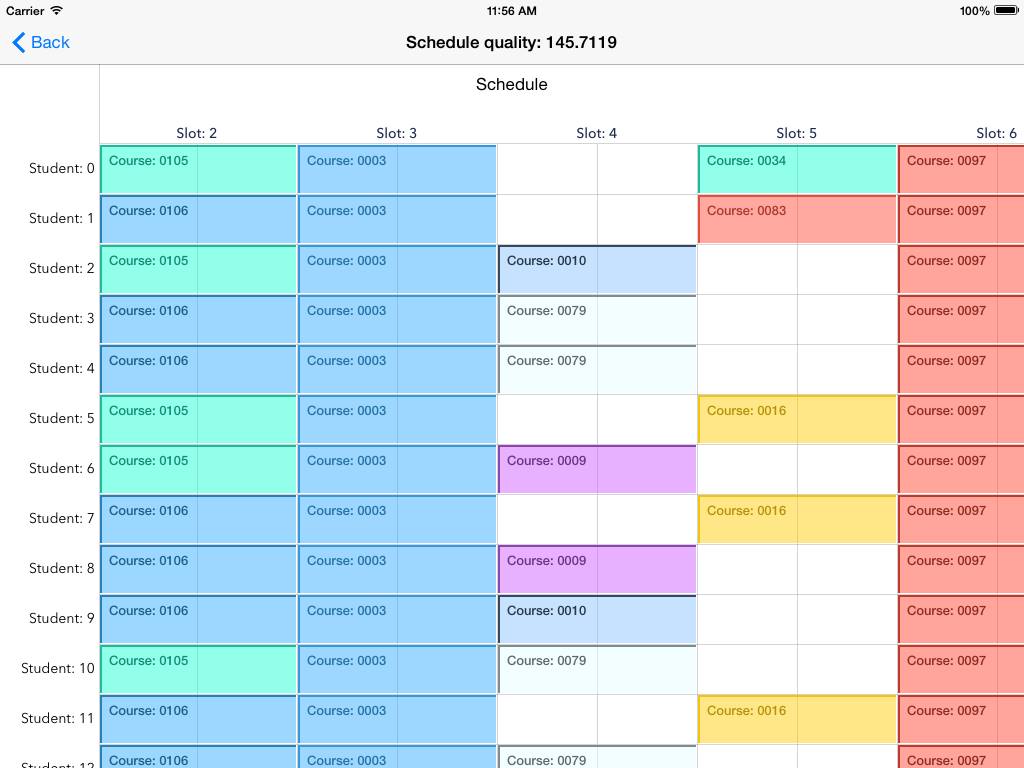
|  |  |
| --- | --- |
| **Metodologia** | **Referencja** |
| Cluster-based/decomposition | Balakrishnan et al. (1992), Burke and Newall (1999), Qu and Burke (2007) |
| Tabu search | Di Gaspero and Schaerf (2001), White and Xie (2001) |
| Great deluge algorithm | Burke et al. (2004b) |
| Ant algorithms | Eley (2007) |
| Genetic algorithms | Burke et al. (1995), Ülker et al. (2007) |
| Hyper-heuristic | Bilgin et al. (2007), Ersoy et al. (2007), Pillay and Banzhaf (2009) |

# Graficzny interfejs użytkownika



Rysunek 1 Ekran Menu

Rysunek 1 przedstawia menu główne programu z dwoma przyciskami. Po naciśnięciu przycisku „Generate Schedule” nastąpi generowanie nowego harmonogramu. Natomiast drugi przycisk odpowiadania za przejście do najlepszego dotychczas wygenerowanego harmonogramu. Natomiast rysunek 2 przedstawia widok harmonogramu. Tutaj widzimy harmonogram dla każdego studenta w odpowiednich przedziałach czasowych. W danym przedziale czasowym student może lub nie mieć zaplanowany odpowiedni kurs. Studenci są reprezentowani przez wiersze harmonogramu ,a przedziały czasowe przez kolumny harmonogramu.



Rysunek 2 Widok harmonogramu

# Diagramy funkcjonowania aplikacji

# Wyniki przeprowadzonych testów

Poprawność przeprowadzonych wyników była zweryfikowana programem[[3]](#footnote-4) autorstwa Rong Qu z Uniwersytetu w Nottingham, który kalkuluje wynik na podstawie gotowego rozwiązania. Testy przeprowadzone były na następujących urządzeniach:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | MacBook Pro 2010 Mid | iPhone 5 | iPad Mini 2 | Mac mini 2012 |
| Procesor | Intel Core 2 Duo 2,4GHz | ARMv7 Apple A6 1.3 GHz | ARM64 Apple A7 1.3 GHz 64-bit | Intel Core i7  quad-core 2.3GHz |
| Liczba rdzeni | 2 | 2 | 2 | 4 |
| Ram | 8 GB | 1GB | 1GB | 16GB |

## Koszt algorytmu



Dla danych wejściowych zostały ustawione następujące wartości:

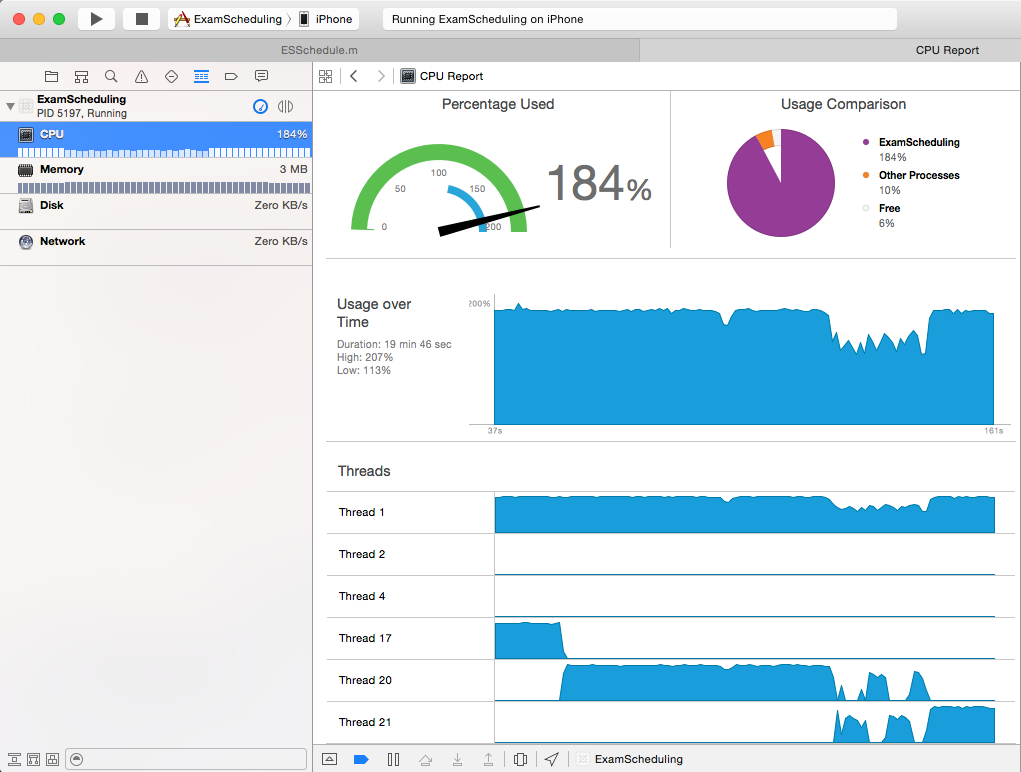
Początkowa temperatura: 1200.95

Temperatura zamarzania: 2^-6

Obniżanie temperatury: 0.95

Zbiór danych, które zostały dostarczone przez prowadzącego zawiera:

Liczba studentów: 611

Liczba kursów: 113

Poniższa tabela pokazuje czas oraz otrzymany wynik algorytmu w zależności od urządzenia na którym wykonywane były testy:

|  |  |  |
| --- | --- | --- |
|  | Czas (s) | Wynik |
| Macbook 2010 Mid | 1570 | 145,711 |
| iPhone 5 | 3620 | 148,947 |
| iPad Mini 2 | 670 | 152,189 |
| Mac Mini 2012 | 420 | 147,685 |

Jak widać czasy generowania algorytmu znacząco się różnią w zależności od urządzenia. Wszystkie urządzenia osiągały wynik pozbawiony konfliktów przy temperaturze w okolicach zera, przykładowo dla iPhone 5 była to temperatura 1,18, dla takiej temperatury wynik wynosił 152.481. Zaskoczony jestem wydajnością iPada Mini 2, wypadł on zdecydowanie szybciej od reszty, nawet od komputera z procesorem C2d, który jak wydawało się powinien być zdecydowanie wydajniejszy od procesorów z rodziny ARM. Algorytm daje bardzo podobne dla każdego uruchomienia. Najlepszy wynik jaki uzyskałem to 145,711.

# Scenariusze testów jednostkowych

Testy w formie testów jednostkowych pokrywają podstawowe scenariusze liczenia punktów karnych. Ze względu na niskopoziomowość tych testów część może wydawać się bezsensowna ,ale tak naprawdę pokrywają one scenariusze skrajne i udowadniają działanie algorytmu dla różnych struktur oraz zestawów danych. W tym dziale zostaną opisane scenariusze testowe. Testy polegają na sprawdzeniu stanów punktów karnych dla studenta za egzaminy obok siebie oraz punktów karnych za równoczesne egzaminy.

## TEST 1 – JEDEN SLOTY W HARMONOGRAMIE

W tym teście ilość time slot-ów jest równe jeden. Ilość punktów karnych w obydwu przypadkach powinna być równa zero. Minimalny

## TEST 2 – DWA SLOTY W HARMONOGRAMIE

Dwa sloty to minimalna ilość slotów by zacząć sporządzać harmonogram. Ponieważ egzamin zajmuje 2 sloty i w tym wypadku wartość kar powinna wynosić zero.

## TEST 3 – TRZY SLOTY w HARMONOGRAMIE BEZ ZACHOWANIA KOLEJNOŚCI

Trzy sloty w kolejności 1,3,2 są przeznaczone na harmonogram. Ze względu ze wszystkie zostaną zajęte to kara za sąsiedztwo powinna być policzona dla każdego następująco:

8 + 16 + 16

8 – za sąsiedztwo 1 i 3

16 – za sąsiedztwo 2 i 3

16 – za sąsiedztwo 1 i 2

## TEST 3 – CZTERY SLOTY w HARMONOGRAMIE BEZ ZACHOWANIA KOLEJNOŚCI

Cztery sloty w kolejności 1,4,2,3 są przeznaczone na harmonogram. Ze względu ze wszystkie zostaną zajęte to kara za sąsiedztwo powinna być policzona dla każdego następująco:

4 + 16 + 8 + 8 + 16 +16

4 – za sąsiedztwo 1 i 4

16 – za sąsiedztwo 1 i 2

8 – za sąsiedztwo 1 i 3

8 – za sąsiedztwo 2 i 4

16 – za sąsiedztwo 2 i 3

16 – za sąsiedztwo 3 i 4

## TEST 4 – SLOT „0” w HARMONOGRAMIE

Trzy sloty w kolejności 1,0,2 są przeznaczone na harmonogram. Ze względu ze wszystkie zostaną zajęte to kara za sąsiedztwo powinna być policzona dla każdego następująco:

16 + 16 + 8

16 – za sąsiedztwo 2 i 3

16 – za sąsiedztwo 1 i 2

8 – za sąsiedztwo 1 i 3

## TEST 4 – SLOT „0” w HARMONOGRAMIE Z DUŻĄ PRZERWĄ

Dwa sloty w kolejności 5,0 są przeznaczone na harmonogram. Ze względu ze wszystkie zostaną zajęte to kara za sąsiedztwo powinna być równa 1.

## TEST 5 – SLOT „0” Z KONFIKTEM

Trzy sloty w kolejności 5,0,5 są przeznaczone na harmonogram. Ze względu ze wszystkie zostaną zajęte to kara za sąsiedztwo powinna być policzona dla każdego następująco:

1 + 1

1 – za sąsiedztwo 0 i 5’

1 – za sąsiedztwo 0 i 5’’

Dodatkowo kara za równoczesne dwa egzaminy powinna być równa 4000000.00

## TEST KONFLIKTU

Dwa sloty w kolejności 5,5 są przeznaczone na harmonogram. Ze względu że to ten sam time slot to nie ma kary za sąsiedztwo. Kara za równoczesne dwa egzaminy powinna być równa 4000000.00.

# Bibliografia

1. <https://github.com/nicklockwood/FastCoding> [↑](#footnote-ref-2)
2. A Construction Approach for Examination Timetabling based on Adaptive Decomposition and Ordering (Syariza Abdul Rahman, Edmund Burke, Andrzej Bargiela, Barry McCollum and Ender Özcan) - <http://zaborowski.me/pracownia/research/d98.pdf> [↑](#footnote-ref-3)
3. Rong Qu, PhD, BSc - <http://www.cs.nott.ac.uk/~rxq/data.htm> [↑](#footnote-ref-4)