Kodowanie Hamminga (semestr letni 2013/2014)

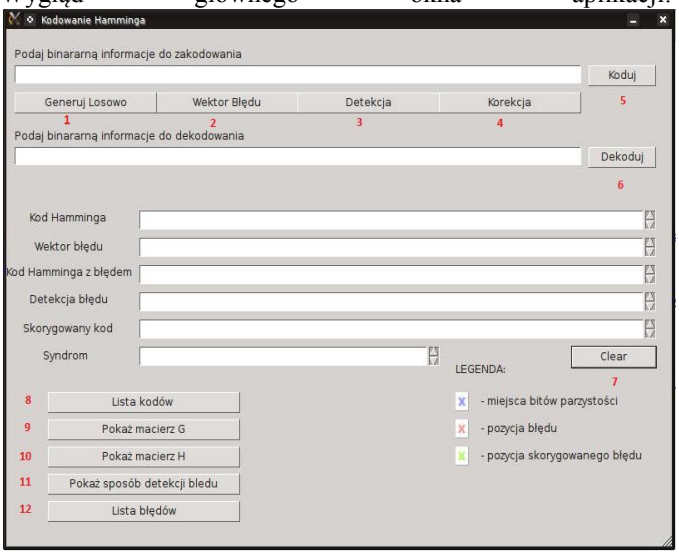
Szymon Tomczyk 259045, Kamil Ruchała 259030

I. Wprowadzenie

Projekt ma na celu zademonstrowanie działania nadmiarowego kodu Hamminga - poprzez dodanie dodatkowych bitów parzystości wykrywa i koryguje przekłamanie jednego bitu w ciągu kodowym. Jest to aplikacja z graficznym interfejsem użytkownika, która pozwala na zaobserwowanie podstawowych własności kodu. Do funkcji programu należą m.in. konstrukcja kodu, kodowanie oraz dekodowanie wprowadzonej wiadomości, prezentacja macierzowego rozwiązania problemu.

II. Instrukcja konfiguracji i użytkowania

Aby włączyć program należy uruchomić skrypt napisany w języku Python. Aplikacja korzysta z dwóch zewnętrznych bibliotek PyQt (odpowiedzialne za GUI) oraz NumPy (ułatwia operowanie na macierzach), obie z nich są niezbędne do poprawnego przeanalizowania skryptu przez interpreter. Wygląd głównego okna aplikacji:



Rys 1 Główne okno programu

Poniżej zostały opisane wszystkie parametry programu, oraz jego funkcjonalność.

1."Generuj Losowo" - za pomocą suwaka generuje losowy ciąg bitów o podanej długości.

2."Wektor Błędu" - symulujemy błąd wprowadzony przez kanał transmisyjny (np. 0001 - błąd na 4 pozycji, 01 - błąd na 2 pozycji).

3."Detekcja" - dla zakodowanego ciągu, sprawdza dzięki właściwościom kodu Hamminga(syndrom) na której pozycji wystąpił błąd.

4."Korekcja" - ściśle powiązana z detekcją, poprawią bit na pozycji zwróconej przez detekcje na przeciwny.

5."Koduj" - dla wpisanego lub wygenerowanego losowo ciągu bitów jest tworzona macierz G i H, oraz ciąg poprzez dodanie bitów parzystości zostaje zakodowany.

6."Dekoduj" - dla podanego zakodowanego ciągu sprawdza jego syndrom, poprawia ewentualny błąd oraz usuwa bity parzystości zostawiając samą informację.

7."Clear" - służy do wyczyszczenia formularzy.

8."Lista kodów" - tworzy listę wszystkich możliwych kodów o takiej samej długości jak długość ciągu wprowadzonego/wygenerowanego losowo przez użytkownika i wyświetlą ją w nowym oknie.

9."Pokaż macierz G" - wyświetla w nowym oknie macierz G wygenerowaną na podstawie wprowadzonego ciągu.

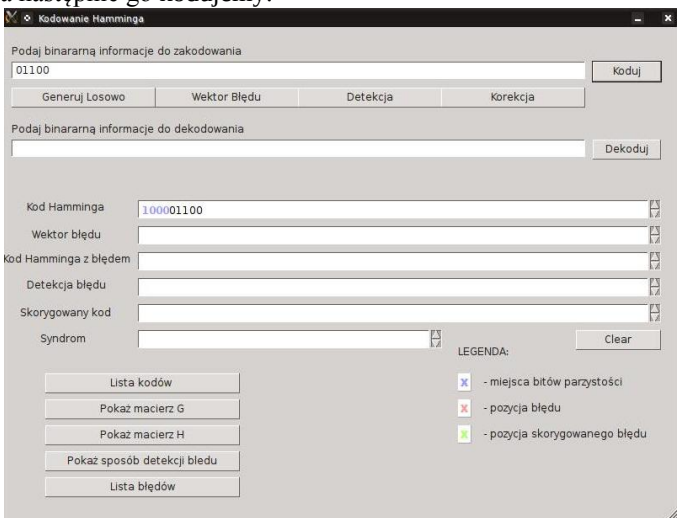
10."Pokaż macierz H" - wyświetla w nowym oknie macierz H wygenerowaną na podstawie wprowadzonego ciągu.

11."Pokaż sposób detekcji błędu" - wyświetla w nowym oknie macierz H oraz syndrom i na tej podstawie prezentuje na której pozycji wystąpił błąd.

12. "Lista błędów" - generuję listę wszystkich możliwych pojedynczych błędów dla zakodowanego ciągu oraz przedstawia je wraz z odpowiadającymi im syndromami.

III. Przykładowe użycie programu

1.Wprowadzamy ręcznie lub generujemy losowo ciąg bitów a następnie go kodujemy:

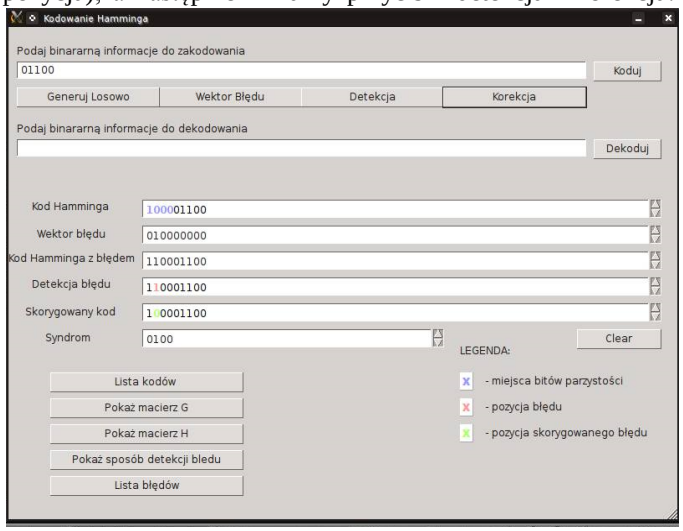


Rys 2 Zakodowany ciąg

W pierwszym formularzu obserwujemy zakodowany ciąg,

bity parzystości są koloru niebieskiego, a bity informacyjne pozostają w kolorze czarnym. Miejsca w których pojawiają sie bity parzystości(na początku ciągu) są uwarunkowane przez implementacje macierzy G oraz H.

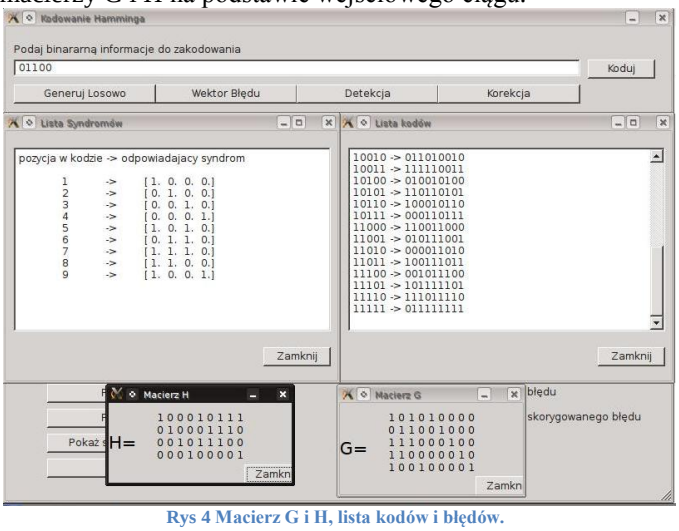
2.Wprowadzamy wektor błędu w tym wypadku 01(druga pozycja), a następnie klikamy przyciski detekcja i korekcja:



Rys 3 Ciąg po detekcji i korekcji

Wektor błędu rozszerza się do długości zakodowanej wiadomości poprzez uzupełnienie zerami. W kolejnych formularzach zostają wyświetlone: Kod Hamminga z błędem naniesionym przez kanał transmisyjny, kod po detekcji i korekcji oraz syndrom błędu - wektor obliczony poprzez pomnożenie macierzy H oraz ciągu kodowego.

3.Kolejne przyciski pozwalają nam zaobserwować właściwości kodu Hamminga, oraz sposób generowania macierzy G i H na podstawie wejściowego ciągu.



Rys 4 Macierz G i H, lista kodów i błędów.

Dane te służą analizie i interpretacji kodu Hamminga. Już na pierwszy rzut oka widać podobieństwo między syndromami dla kolejnych błędów a macierzą H.

IV. Wnioski

Kod Hamminga ma szerokie zastosowanie w przemyśle informatycznym, głównie ze względu na jego prostotę. Implementuje się go m.in. w pamięciach RAM czy płycie CD w celu wykrycia błędów (np. podczas zarysowania). Projekt miał na celu zobrazowanie konstrukcji kodu dla dowolnego ciągu binarnego, oraz zaprezentowanie macierzowego podejścia do problemu. Bity parzystości mogłyby również być wyliczanie na podstawie bramek XOR, na tej samej podstawie mogłaby odbywać się detekcja i korekcja błędów. Reprezentacja kodu Hamminga za pomocą macierzy niesie ze sobą wiele korzyści:

-zmieniając macierz G, manipulujemy miejscami na których pojawia się bity informacyjne w ciągu. Kod dalej posiada te same właściwości.

-zakodowanie wiadomości sprowadza się tylko do pomnożenia macierzy generującej i ciągu wejściowego, jest to znacznie szybsze i prostsze niż wyliczanie kolejnych sum modulo 2.

-korekcja i dekodowanie wiadomości jest równie proste jak jej kodowanie. Po pomnożeniu zakodowanego ciągu przez macierz H otrzymujemy syndrom, na podstawie którego wskazujemy na przekłamany bit - zostało to bardzo dobrze zobrazowane w programie.

Istnieje możliwość implementacji rozszerzonego kodu Hamminga, który za pomocą dodatkowego bitu jest w stanie wykryć do 2 błędów zakodowanym ciągu.