**KMP** – od **1**, wartość = **k=1,2**.., przesunięcie o **index (last ok) – wartość**  
**BMH** – od **0** TOH\* **lenght-1-index**, przesunięcie o **last górny**  
**LCS** – **0,0, \+1**, **max(ab**), lewa, góra, przekątna  
**Lev** – **0,1,2, \, min+1**, jak najszybciej, w lewo **delete**  
**Zipf rank\*częstość**=stałą  
**Heaps** V(n)=k\*n^b **pierw kwadr**  
**Page rank** – jakość tekstu proporcjonalna do **wskazań** na niego  
**Iton a>b** typ A  
**Hamming** ilość operacji żeby zmienić **d(s,s) =2**  
**Full text** wskazuje każdą pozycję wystąpienia  
**Inverted index** – **słowa** i ich **wystąpienia** (pozycje) w **bazie**  
**K+1** - **k części bez błędów**  
**Patritia tree** – **połączenie** **noda** bez gałęzi z **dzieckiem**   
użycie 1 **bitu** na **wskaźnik** lub 1 **wskażnika** na all **dzieci**  
**Rabin karp ###  
DP** duży problem na **małe**, **rozwiązanie w pamięci** i wykorzystuje

|  |
| --- |
| **KMP** – od **1**, wartość = **k=1,2**.., przesunięcie o **index (last ok) – wartość** **BMH** – od **0** TOH\* **lenght-1-index**, przesunięcie o **last górny** **LCS** – **0,0, \+1**, **max(ab**), lewa, góra, przekątna **Lev** – **0,1,2, \, min+1**, jak najszybciej, w lewo **delete** **Zipf rank\*częstość**=stałą **Heaps** V(n)=k\*n^b **pierw kwadr** **Page rank** – jakość tekstu proporcjonalna do **wskazań** na niego **Iton a>b** typ A **Hamming** ilość operacji żeby zmienić **d(s,s) =2** **Full text** wskazuje każdą pozycję wystąpienia **Inverted index** – **słowa** i ich **wystąpienia** (pozycje) w **bazie** **K+1** - **k części bez błędów** **Patritia tree** – **połączenie** **noda** bez gałęzi z **dzieckiem**  użycie 1 **bitu** na **wskaźnik** lub 1 **wskażnika** na all **dzieci** **Rabin karp ### DP** duży problem na **małe**, **rozwiązanie w pamięci** i wykorzystuje |
| **KMP** – od **1**, wartość = **k=1,2**.., przesunięcie o **index (last ok) – wartość** **BMH** – od **0** TOH\* **lenght-1-index**, przesunięcie o **last górny** **LCS** – **0,0, \+1**, **max(ab**), lewa, góra, przekątna **Lev** – **0,1,2, \, min+1**, jak najszybciej, w lewo **delete** **Zipf rank\*częstość**=stałą **Heaps** V(n)=k\*n^b **pierw kwadr** **Page rank** – jakość tekstu proporcjonalna do **wskazań** na niego **Iton a>b** typ A **Hamming** ilość operacji żeby zmienić **d(s,s) =2** **Full text** wskazuje każdą pozycję wystąpienia **Inverted index** – **słowa** i ich **wystąpienia** (pozycje) w **bazie** **K+1** - **k części bez błędów** **Patritia tree** – **połączenie** **noda** bez gałęzi z **dzieckiem**  użycie 1 **bitu** na **wskaźnik** lub 1 **wskażnika** na all **dzieci** **Rabin karp ### DP** duży problem na **małe**, **rozwiązanie w pamięci** i wykorzystuje |
| **KMP** – od **1**, wartość = **k=1,2**.., przesunięcie o **index (last ok) – wartość** **BMH** – od **0** TOH\* **lenght-1-index**, przesunięcie o **last górny** **LCS** – **0,0, \+1**, **max(ab**), lewa, góra, przekątna **Lev** – **0,1,2, \, min+1**, jak najszybciej, w lewo **delete** **Zipf rank\*częstość**=stałą **Heaps** V(n)=k\*n^b **pierw kwadr** **Page rank** – jakość tekstu proporcjonalna do **wskazań** na niego **Iton a>b** typ A **Hamming** ilość operacji żeby zmienić **d(s,s) =2** **Full text** wskazuje każdą pozycję wystąpienia **Inverted index** – **słowa** i ich **wystąpienia** (pozycje) w **bazie** **K+1** - **k części bez błędów** **Patritia tree** – **połączenie** **noda** bez gałęzi z **dzieckiem**  użycie 1 **bitu** na **wskaźnik** lub 1 **wskażnika** na all **dzieci** **Rabin karp ### DP** duży problem na **małe**, **rozwiązanie w pamięci** i wykorzystuje |