# Wyszukiwanie i Przetwarzanie Informacji WWW

## Automatyczne zbieranie dokumentów WWW
## 2: Zagadnienia techniczne i przechowywanie

Marcin Sydow

PJWSTK

# Plan dzisiejszego wykładu:

Automatyczne Zbieranie Kolekcji Dokumentów WWW.

- Wprowadzenie
- Zapewnienie świeżości kolekcji
- Zagadnienia współbieżności przy zbieraniu
- Repozytorium dokumentów

# Moduł Zbierania - zasada działania (przypomnienie)

**Startowy** zestaw adresów URL - **inicjalizuje** kolejkę

Dla każdego URL:

1. ściągnąć
2. wyparsować następne adresy URL
3. dorzucić je do kolejki

Postępować tak, aż do zajdą **warunki zakończenia** (np. wyczerpanie zasobów)

Następnie (niezależnie od procesu zbierania):

1. ściągnięte dokumenty trafiają do **Repozytorium**
2. są później podawane do Modułu Indeksującego - powstaje **Indeks**

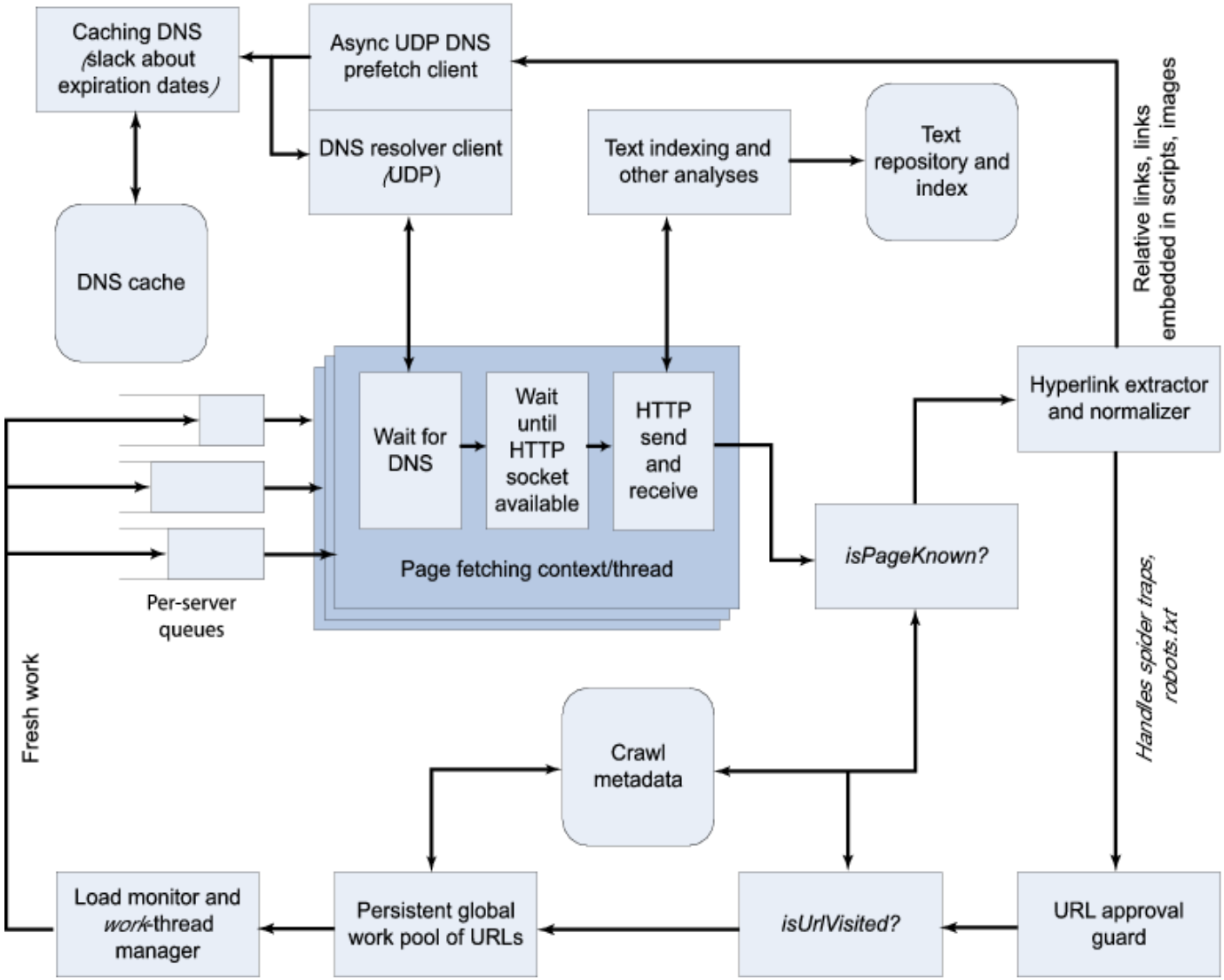
# Modyfikacje Podstawowego Algorytmu

Istnieją różne warianty podstawowego zachowania Modułu Zbierania, w zależności od jego przeznaczenia.

Przykłady:

- zbieranie dokumentów z jak **największej liczby hostów** w danym uniwersum (np. domena .pl)
- zbieranie **maksymalnej liczby dokumentów** z określonej grupy hostów lub domen
- zbieranie **"tematyczne"** (ang. focused crawling)

# Architektura Zbieracza




(wg. "Mining the Web" S.Chakrabarti, Morgan-Kaufmann, 2003)

# Strategia Odświeżania

Strony WWW zmieniają się.

WWW jest nie tylko bardzo dynamiczny, ale również **zróżnicowany** - niektóre strony zmieniają się w każdej minucie (np. wiadomości) inne co kilka miesięcy

Należy ustalić **strategię ponownego odwiedzania** stron. Np. częstotliwość ponownego odwiedzania stron może być:

- jednorodna (jednakowa częstotliwość odświeżania wszystkich stron)
- proporcjonalna (do częstotliwości zmian odwiedzanych stron - bazuje jedynie na oszacowaniu)
- specjalna (ustalana odgórnie dla pewnych grup dokumentów. Np. serwisy finansowe i wiadomości - codziennie)

Powyższe (i inne) strategie odświeżania można stosować równocześnie do różnych grup dokumentów.

# Przykładowe najprostsze miary świeżości

Najprostszy przykład: Dyskretna miara świeżości dokumentu $d_i$ w momencie t:

$$F(d_i, t) = \begin{cases} 1 & d_i \text{ aktualne w momencie } t \\ 0 & w.p.p. \end{cases}$$

Świeżość w momencie t kolekcji K zawierającej N dokumentów można zdefiniować następująco:

$F(K, t) = \frac{1}{N} \sum_{i=1}^{N} F(d_i, t)$

(wg. J.Cho, "Crawling the Web. Discovery and maintenance of large-scale Web Data, Ph.D. thesis, Stanford Univ.)

# Wiek dokumentu i kolekcji

Przykład miary wieku dokumentu $d_i$ w momencie t o czasie ostatniej modyfikacji $mod(d_i)$:

$$A(d_i, t) = \begin{cases} 0 & d_i \text{ aktualne w momencie } t \\ t - mod(d_i) & w.p.p. \end{cases}$$

Wiek kolekcji K zawierającej N dokumentów, w momencie t:
$A(K, t) = \frac{1}{N} \sum_{i=1}^{N} A(d_i, t)$
(wg. J.Cho, "Crawling the Web. Discovery and maintenance of large-scale Web Data, Ph.D. thesis, Stanford Univ.)

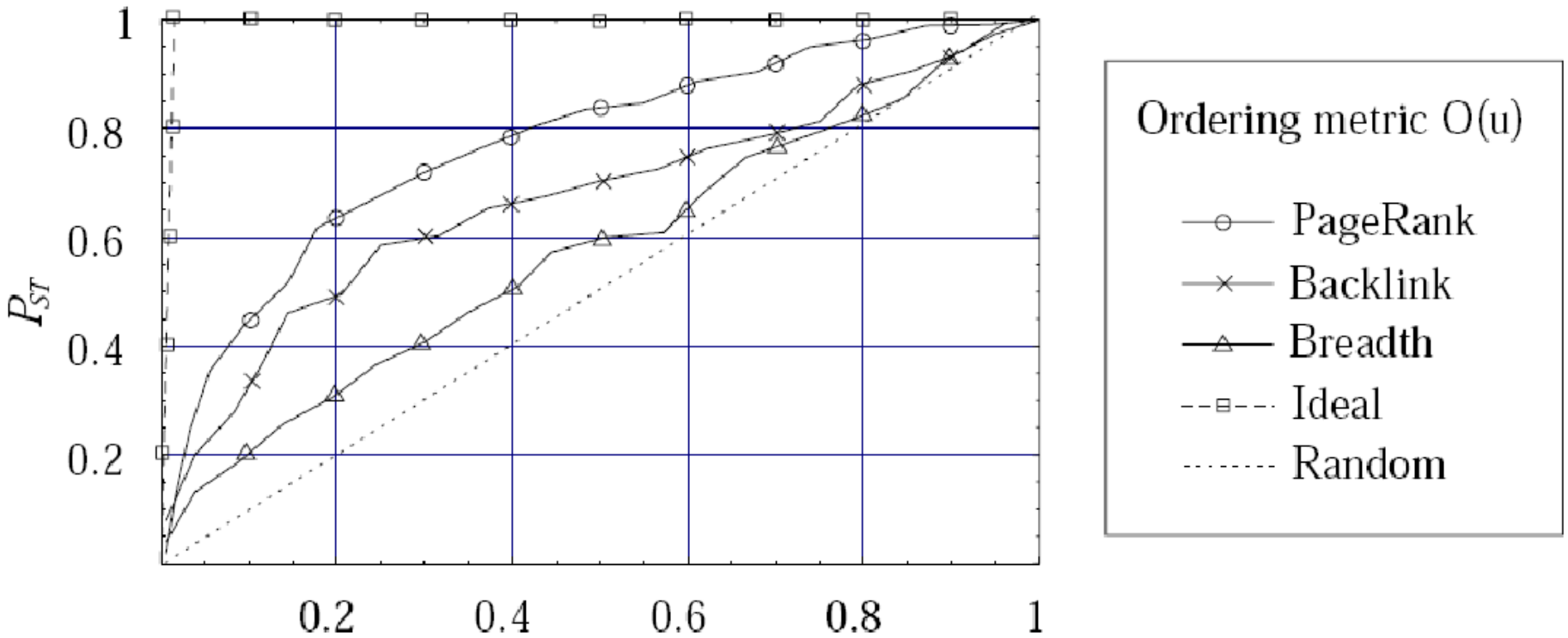
# Dynamiczna Miara Świeżości

Można też definiować miarę świeżości z uwzględnieniem upływu czasu t:
dla dokumentu $d_i$:
$\overline{F}(d_i) = \lim_{t\to\infty} \frac{1}{t} \int_0^t F(d_i, t)dt$

i dla całej kolekcji K:
$\overline{F}(K) = \lim_{t\to\infty} \frac{1}{t} \int_0^t F(K, t)dt$

(wg. J.Cho, "Crawling the Web. Discovery and maintenance of large-scale Web Data, Ph.D. thesis, Stanford Univ.)

# Porównanie strategii kolejności zbierania




Porównanie tempa lokalizowania stron wartościowych przy użyciu różnych miar ważności stron. $P_{ST}$ - udział zebranych stron "najważniejszych" (wg. J.Cho, "Crawling the Web. Discovery and maintenance of large-scale Web Data, Ph.D. thesis, Stanford Univ.)

# Wybór strategii odświeżania

Udowodniono, że przy powyżej zdefiniowanych miarach i założeniu, iż dokumenty są modyfikowane zgodnie z procesem Poissona strategia jednorodna jest zawsze lepsza od strategii proporcjonalnej.

Pokazano też optymalną (w sensie maksymalizacji świeżości kolekcji) strategię odświeżania przy założeniu stałych częstotliwości odświeżania dokumentów.

(J.Cho et al. "Synchronizing a database to improve freshness. Proc. of the ICMD, 2000)

# Od koncepcji do szczegółów

Nadmienione poprzednio zagadnienia są natury wysoko-poziomowej i koncepcyjnej.

Mimo prostej podstawowej zasady działania, moduł zbierający (szczególnie dotyczący systemów średniej i dużej skali) jest również związany z szeregiem ciekawych (i często niełatwych) problemów natury technicznej.

W praktyce, to właśnie **zagadnienia techniczne** sprawiają, że zadanie automatycznego zbierania średnich i dużych kolekcji jest uważane za **skomplikowane** i obfitujące w szereg wyzwań natury inżynieryjnej.

Omówione teraz zostaną najważniejsze w/w aspekty techniczne

# Współbieżność Zbierania

Z perspektywy technicznej, podstawowa pojedyncza iteracja modułu zbierającego składa się z następujących kroków:

1. rozwiązywanie DNS (dla hosta w bieżącym adresie URL)
2. nawiązywanie połączenia (przez gniazdo sieciowe) z wyznaczonym serwerem HTTP i wysłanie żądania
3. odbieranie żądanych zasobów (w odpowiedzi)

Ponieważ 2 pierwsze kroki sprowadzają się do **oczekiwania**, system zbierający powinien wykonywać **wiele** takich iteracji **współbieżnie**, aby osiągnąć zadowalającą **szybkość** ściągania, poprzez **przeplatanie** operacji i oczekiwania

# "Wąskie Gardło"

Zauważmy, że dla krótkich dokumentów rozwiązywanie DNS i nawiązywanie połączenia sieciowego z serwerem stanowi stosunkowo dużą część czasu potrzebnego na obsłużenie danego adresu URL.

Dodatkowo, istotną obserwacją jest, że ograniczenia te **nie mogą** być zrównoważone przez zwiększenie przepustowości łącza sieciowego.

Cykl ściągania pojedynczego dokumentu na ogół zakodowany jest za pomocą specjalnego **wątku logicznego** (niezależnego od wątków dostarczanych przez system operacyjny lub środowisko) reprezentującego kolejne poszczególne stany ściągania danego dokumentu.

# Techniczne Wyzwania Dużej Skali

Należy podkreślić, że w przypadku zbierania kolekcji dużej skali, następujące zagadnienia techniczne nabierają **krytycznej** istotności dla prawidłowego działania systemu:

- opóźnienia wynikające z operacji sieciowych sprawiają, że systemy zbierające dużej skali muszą dokonywać ściągania **setek lub tysięcy** dokumentów **równolegle** lub **współbieżnie**
- warunkiem koniecznym wysokiej współbieżności ściągania jest **współbieżność rozwiązywania DNS** (można ją poprawić m.in. przez replikację serwerów DNS)
- wątki, procesy lub "procesy lekkie" zapewniane przez system operacyjny na ogół nie są wystarczająco efektywnym środkiem zapewnienia współbieżności. Należy jawnie zakodować "wątki logiczne" reprezentujące kolejne stany ściągania dokumentu i używać **asynchronicznego** (nieblokującego) interfejsu we/wy
- dużo uwagi należy poświęcić unikaniu ściągania duplikatów i radzenia sobie z "pułapkami"

# Zagadnienia DNS

Zbieranie kolekcji WWW na średnią i dużą skalę stwarza warunki, w których standardowe rozwiązania DNS mogą okazać się niewystarczające.

W większości zastosowań, kiesa DNS (ang. DNS cache) przystosowana jest do przechowywania dostatecznie dużej ilości ostatnio rozwiązywanych adresów.

Jednak zbieracz musi szczególnie unikać obciążania pojedynczych serwerów HTTP, a więc pojawia się wyraźna **nielokalność** odwołań do kiesy DNS. Ponadto, przy zbieraniu dużej i średniej skali występują setki albo tysiące odwołań w ciągu sekundy. Z tych powodów, standardowe rozwiązania DNS mogą okazać się "wąskim gardłem" systemu.

Ponadto, niektóre standardowe implementacje klientów DNS, nie wspierają współbieżnego przetwarzania żądań DNS.

# Rozwiązania Problemów z DNS

Żeby rozwiązać w/w problemy, w systemach dużej skali stosuje się specjalne rozwiązania DNS:

- specjalne klienty DNS
- specjalne serwery kiesy (ang. caching server)
- "odpytywanie wyprzedzające" DNS (ang. DNS prefetching) realizowane przez specjalnego klienta (przez UDP)
- replikacja w/w modułów

Np. w systemie "Mercator" (Compaq System Research Center) zmniejszono czas spędzany przez zbieracza na rozwiązywaniu DNS **z 70% do 14%**. ("High-Performance Web Crawling", M.Najork, A.Heydon)

# Fazy cyklu życia dokumentu

1. Ściąganie
2. Indeksowanie
3. Obsługa zapytań
4. Aktualizacja

# Cykl życia dokumentu: faza 1 - zbieranie

Ściąganie dokumentu:

1. URL wskazujący na dokument trafia do kolejki zbieracza
2. dokument jest ściągany przez moduł zbierający
3. parsowanie w celu wydobycia linków (i ew. innych statystyk)
4. kompresja
5. zapisanie w **repozytorium**

# Cykl życia dokumentu: faza 2 - indeksowanie

Indeksowanie:

1. pobranie z **repozytorium**
2. dekompresja
3. preprocessing
4. indeksowanie
5. dokument jest zachowywany w **indeksie** w nowej formie: **list inwersyjnych**

# Dla pełności: pozostałe fazy życia dokumentu

Obsługa zapytania:

- oblicz "otaczający tekst" (ang. snippet)
- w miarę potrzeby zwróć kopię (z **repozytorium**)

Aktualizacja:

- Usuń dokument z **repozytorium** (w momencie ściągnięcia nowszej kopii)

# Przechowywanie Kolekcji - Repozytorium

Dokumenty kolekcji ściąganej przez moduł zbierający trafiają do Repozytorium. Dokumenty są przechowywane na ogół w formie **skompresowanej** na nośnikach dyskowych lub czasami taśmowych.

Repozytorium ma za zadanie zorganizować trwałość i wygodny dostęp do ogromnej ilości danych (dokumentów)

Wymagania te sprawiają, że Repozytorium dokumentów pełni podobne funkcje do systemów plików lub baz danych. Jednak są pewne różnice:

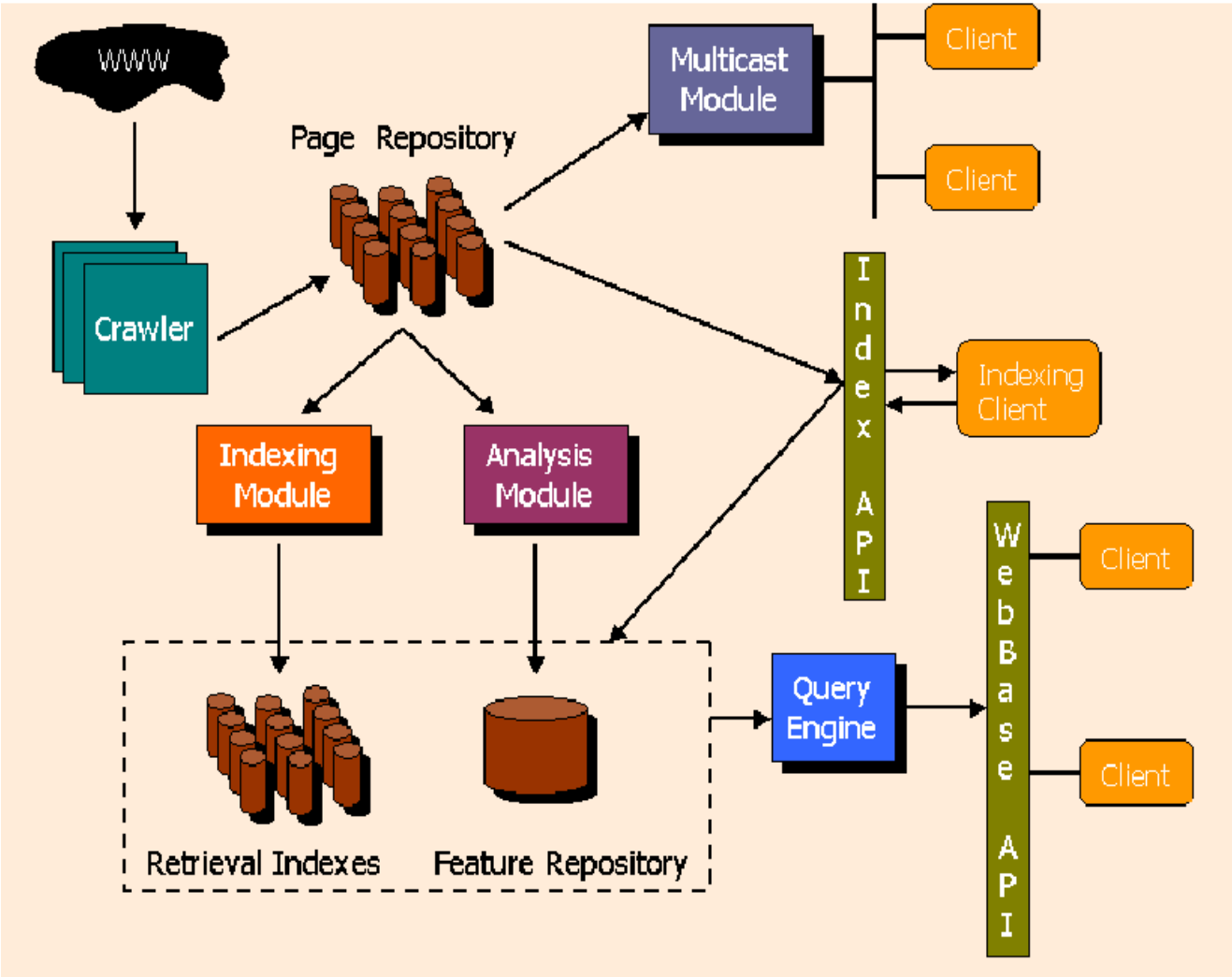
1. Repozytorium **nie musi** zapewniać wspierania transakcji, logowania czy struktury katalogów
2. Z drugiej strony, Repozytorium wyszukiwarki **musi** spełniać pewne dodatkowe wymogi, nietypowe dla systemu bazy danych czy systemów plików

# Cechy Repozytorium

Od repozytorium oczekuje się następujących specjalnych własności:

- Skalowalności
- **Dwóch trybów** dostępu:
  - swobodnego (np. do obsługi zapytań, analizy kolekcji)
  - strumieniowego (np. do szybkiego indeksowania)
- Możliwości masowego uaktualniania kolekcji
- Zarządzania dokumentami nieaktualnymi

# Przykład Architektury - Stanford WebBase




(wg dokumentacji Stanford WebBase)

# Zagadnienia do Rozwiązania

Na ogół, ze względu na rozmiary kolekcji i wydajność, repozytorium ma architekturę **rozproszoną** - kolekcja przechowywana jest w wielu **węzłach** połączonych w sieć.

W takim wypadku, przy projektowaniu repozytorium należy podjąć m.in. następujące decyzje:

- Strategia przydziału dokumentów do poszczególnych węzłów
- Fizyczna organizacja danych na każdym pojedynczym węźle
- Strategia uaktualniania kolekcji

# Strategia rozpraszania dokumentów

Dokumenty można przydzielać do węzłów wg różnych kryteriów, np.:

- losowo (obciążenie węzłów proporcjonalne do ich możliwości)
- w oparciu o obliczony identyfikator (ang. hash) dokumentu. Identyfikator taki może odzwierciedlać rozmaite informacje, np. host z jakiego pochodzi (tzn. dokumenty pochodzące z tego samego hosta mają np. identyczny przedrostek identyfikatora)

W rzeczywistych systemach częstym rozwiązaniem jest także **replikacja** kolekcji.

# Fizyczna organizacja w ramach węzła

Sposób składowania dokumentów musi odzwierciedlać podstawowe operacje, które mogą być wykonywane na repozytorium - dodawanie pojedynczego dokumentu, szybki transfer dużych ilości dokumentów, swobodny dostęp do danego dokumentu.

Do najprostszych metod organizacji należą:

- sekwencyjna (zgodna z kolejnością zapisu)
- mieszająca (ang. hash)

W przypadku organizacji sekwencyjnej utrzymuje się na ogół dodatkową strukturę **indeksu** (np. w postaci B-drzewa) umożliwiającą operację dostępu swobodnego.

Organizacja sekwencyjna mimo swej prostoty spisuje się całkiem nieźle w praktyce (za wyjątkiem masowej obsługi operacji dostępu swobodnego). Nie nadaje się najlepiej do operacji uaktualniania stron.

# Strategia uaktualniania kolekcji

System zbierający może być przystosowany do działania w jednym z dwóch trybów:

- "dyskretnym" (cykliczne ściąganie z "warunkiem zakończenia")
- ciągłym (nieustanne odnawianie kolekcji)

W trybie "dyskretnym" pracują np. niewielkie zbieracze eksperymentalne, naukowe lub zbieracze wyszukiwarek tematycznych, albo związanych z danym portalem.

Tryb ciągły jest natomiast charakterystyczny dla wyszukiwarek komercyjnych wielkiej i średniej skali.

# Strategia uaktualniania kolekcji, c.d.

Przy zbieraniu w trybie dyskretnym można:

- **całkowicie** zastąpić całą poprzednią kolekcję nowymi dokumentami
- uaktualniać tylko **część** dokumentów (np. te o większej wartości)

W przypadku częściowej modyfikacji kolekcji wygodnie jest utrzymywać 2-poziomową strukturę repozytorium (dokumenty "świeże" i "poprzednie"). Ułatwia to wykonywanie na repozytorium **równocześnie** operacji zapisu i odczytu (np. rozwiązanie takie zastosowano w Stanford WebBase)

# Aktualizacja: Klasy dokumentów

Przy rozważaniu problemu aktualizacji kolekcji można rozróżnić 3 klasy dokumentów:

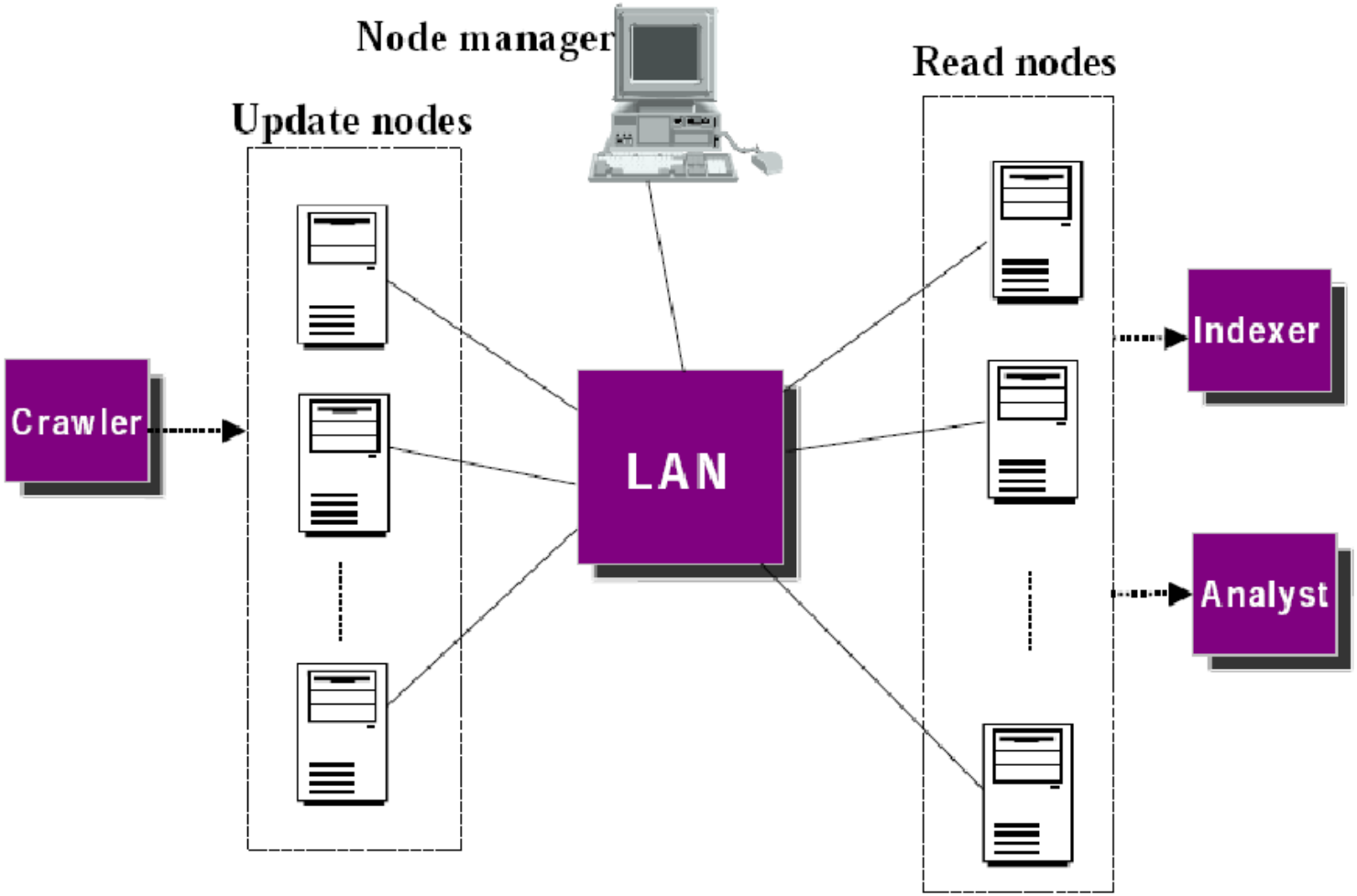
- klasa A: nowo ściągnięte dokumenty (niewidziane wcześniej, lub widziane ale w innej, starszej wersji)
- klasa B: niezmienione kopie dokumentów (w najnowszym cyklu ściągania nie natrafiono na ich nowsze wersje)
- klasa C: stare wersje dokumentów (nowsze wersje już są ściągnięte)

# Kolejność aktualizacji repozytorium

Odświeżenie kolekcji ma więc następującą postać:

1. przyjęcie stron klasy A do repozytorium
2. odbudowanie struktur repozytorium na podstawie dokumentów klas A i B
3. usunięcie dokumentów klasy C

# Przykładowa Architektura Repozytorium




Architektura Repozytorium Stanford WebBase
(wg. "Searching the Web", A.Arasu et al.)

# Przykłady znanych repozytoriów

(poza wyszukiwarkami)

- Stanford WebBase (pionierskie, obecnie mało aktywne?)
- Internet Archive (**poważna sprawa** - obecnie: 150.000.000.000 dokumentów(!) w tym unikalna możliwość obejrzenia jak wyglądał dany dokument w przeszłości (!) - `wayback machine`)

# Na zaliczenie tego wykładu:

1. zagadnienia strategii zbierania i odświeżania
2. (*) miary świeżości
3. protokoły wykluczania
4. (*) współbieżność
5. (*) zagadnienia DNS
6. Cykl życia dokumentu
7. Do czego i kiedy używane jest repozytorium?
8. Pożądane cechy repozytorium
9. Strategie odświeżania

# Dziękuję za uwagę

Dziękuję za uwagę.