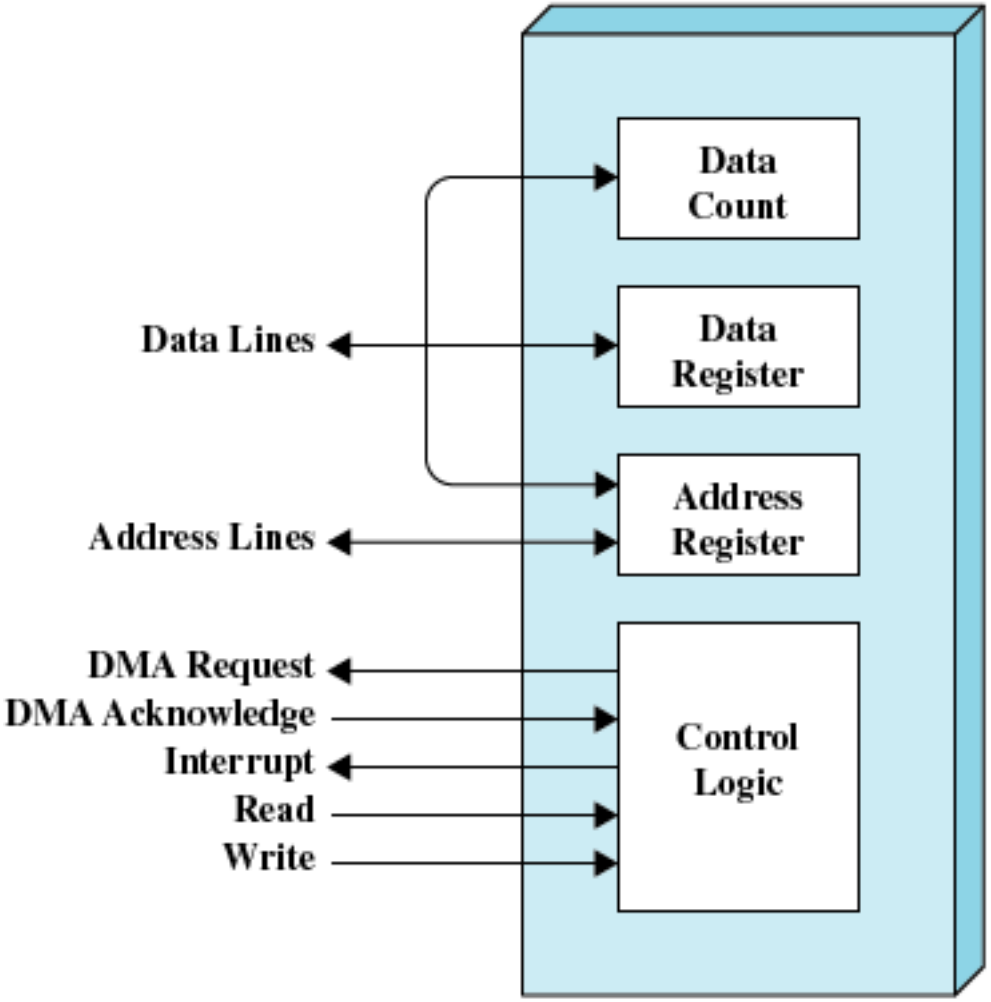
Systemy wejścia/wyjścia

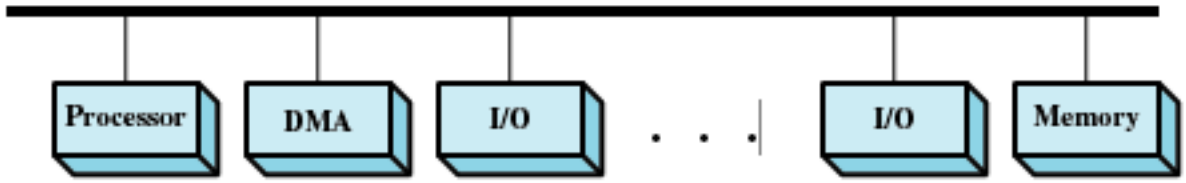
- programowane operacje wejścia/wyjścia:
  proces inicjuje i nadzoruje transfer I/O — odpytywanie (*polling*)
- operacje wejścia/wyjścia sterowane przerwaniami:
  procesor inicjuje transfer I/O, po czym oczekuje na przerwanie z modułu I/O;
  w tym czasie procesor może wykonywać inne procesy, a nawet ten sam proces, jeśli nie wymaga on oczekiwania na zakończenie operacji
- bezpośredni dostęp do pamięci:
  główny procesor tylko inicjuje, a dedykowany procesor DMA wykonuje operacje I/O

	No Interrupts	**Use of Interrupts**
I/O-to-memory transfer through processor	Programmed I/O	Interrupt-driven I/O
Direct I/O-to-memory transfer		Direct memory access (DMA)

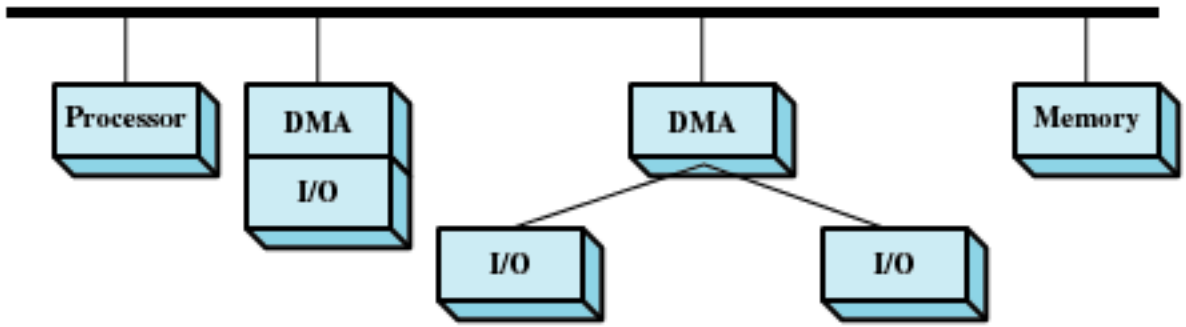
Kanał DMA


Data
Count
Data Lines
Data
Register
Address
Register
Address Lines
DMA Request
DMA Acknowledge
Interrupt
Read
Write
Control
Logic

Konfiguracje kanałów DMA

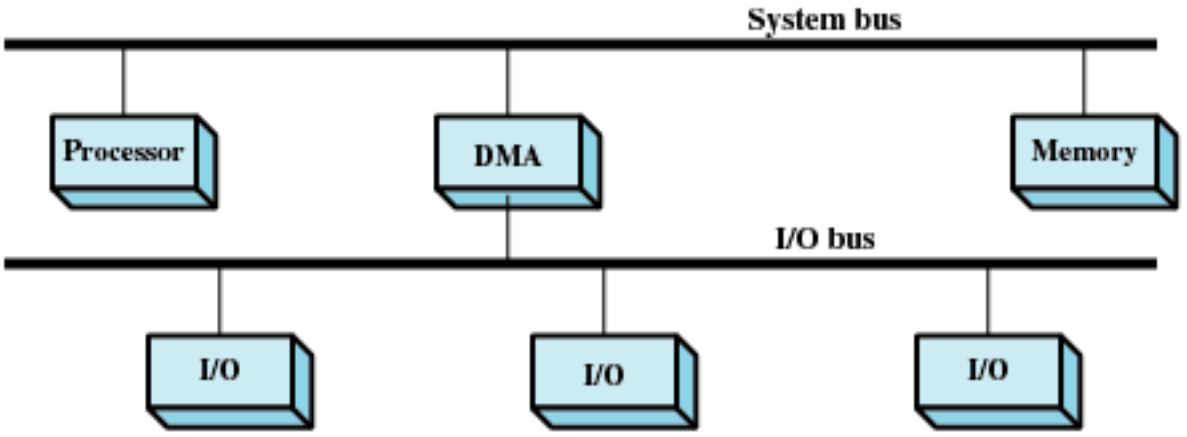


(a) Single-bus, detached DMA



(b) Single-bus, Integrated DMA-I/O

Konfiguracje DMA (cd.)



(c) I/O bus

Krótkie podsumowanie — pytania sprawdzające

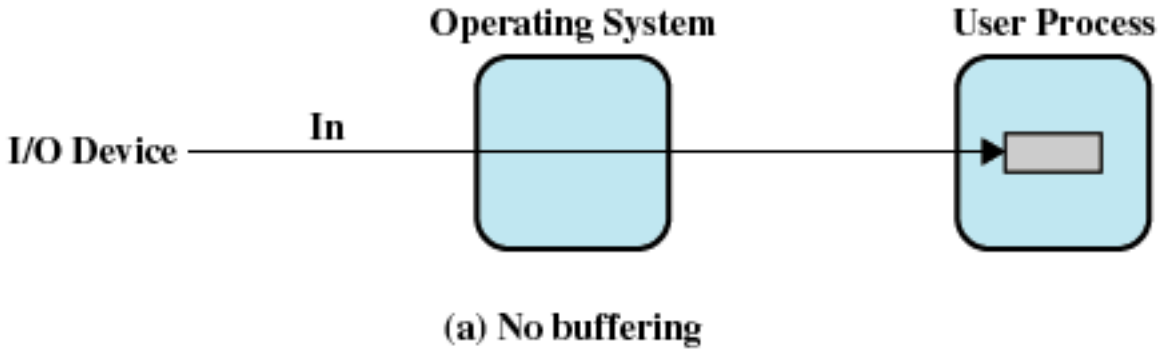
1. Jakie są trzy główne podejścia do implementacji operacji wejścia/wyjścia?

2. Jakie korzyści daje implementacja I/O wykorzystująca przerwania?

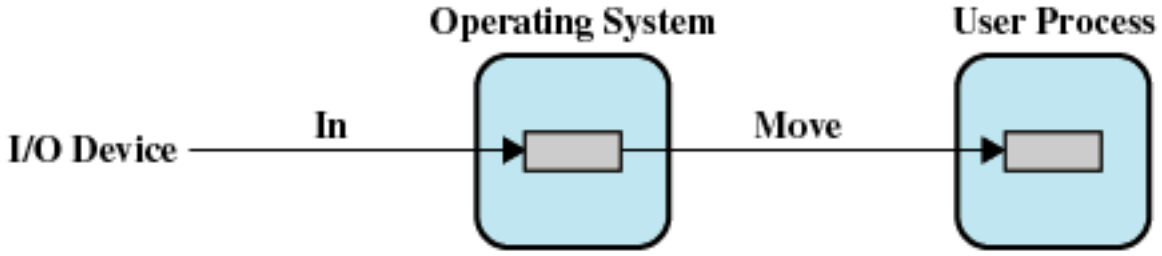
3. Jakie korzyści dają kanały DMA w implementacji operacji I/O?

Buforowanie operacji I/O

Bufory są obszarami pamięci przydzielanymi w celu tymczasowego przechowania danych transferowanych pomiędzy urządzeniami, albo między urządzeniem a programem.
Buforowanie pozwala oddzielić operacje I/O od wykonywania kodu procesu.
W szczególności ma ono znaczenie w powiązaniu ze stronicowaniem pamięci wirtualnej.



(a) No buffering



(b) Single buffering

Zastosowania buforowania

Buforowanie stosuje się w następujących typowych sytuacjach:

- w celu dostosowania różnych prędkości źródła i odbiorcy danych, np. z klawiatury do programu, z portu komunikacyjnego do pliku dyskowego, itp.

- w celu dostosowania przesyłania między urządzeniami posługującymi się jednostkami transferu różnej wielkości, np. przy transmisjach sieciowych fragmentacja większego komunikatu na mniejsze pakiety sieciowe i rekonstrukcja (*reassembly*) oryginalnego komunikatu po jego odebraniu

- w celu zaimplementowania **semantyki kopii** (*copy semantics*) przy transferach

  Np. przy wysyłaniu pliku na drukarkę, co powinno się stać jeśli po zainicjowaniu drukowania program (lub użytkownik) nadpisze dalszy fragment pliku? Semantyka kopii określa, że wydrukowana powinna zostać oryginalna wersja pliku; można to osiągnąć przez jej skopiowanie do bufora w momencie zainicjowania drukowania.

Buforowanie dla urządzeń blokowych i strumieniowych

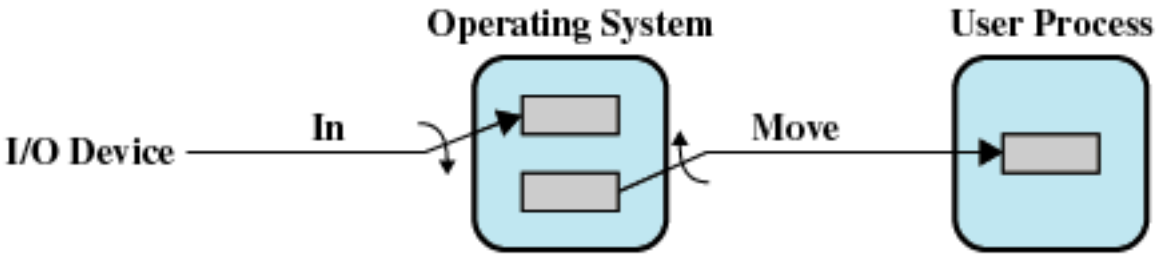
Buforowanie ma różny charakter dla różnych urządzeń I/O:

- urządzenia blokowe: dyski, taśmy — bloki ustalonego rozmiaru, transfer odbywa się blokami,

- urządzenia strumieniowe: terminale, drukarki, skanery, porty komunikacyjne, modemy, myszy, itp., — transfer odbywa się jednostkami o zmiennej długości.

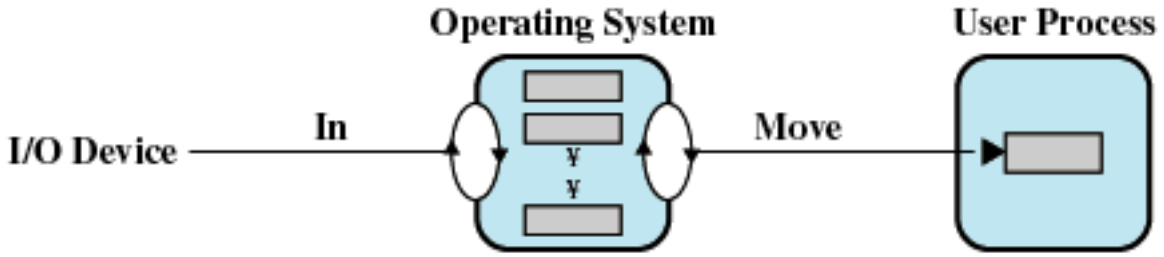
Podwójne i wielostopniowe buforowanie, bufory kołowe

Gdy proces wykonuje jeden transfer za drugim, korzystne okazuje się **podwójne buforowanie**. Poza zwykłym zwiększeniem pojemności buforów, pozwala to odizolować operacje transferu z urządzenia do bufora systemowego, od transferu z bufora systemowego do przestrzeni użytkownika.



(c) Double buffering

Dla jeszcze dłuższych transferów można stosować **buforowanie wielostopniowe**, które wykorzystuje strukturę **bufora kołowego** (*circular buffer*).



(d) Circular buffering

Krótkie podsumowanie — pytania sprawdzające

1. Jaki jest główny cel buforowania operacji wejścia/wyjścia?

2. Wymień typowe sytuacje wymagające buforowania.

3. Czym różni się buforowanie pojedyncze od podwójnego?

4. Czy podwójne buforowanie pozwala zwiększyć prędkość transferów I/O?

Macierze RAID

Pod koniec lat 80-tych XX wieku powstała koncepcja budowy macierzy dyskowych sformalizowana jako RAID (*Redundant Array of Inexpensive Disks*).

Specjalny sterownik obsługuje macierz dysków realizując jeden z szeregu istniejących algorytmów RAID, zapisując i odczytując dane na/z różnych dysków w celu:

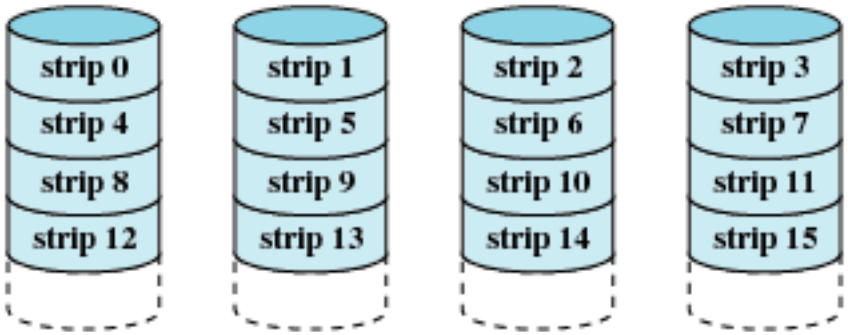
- zwiększenia przepustowości systemu,
- zwiększenia niezawodności systemu i odporności na awarie poszczególnych dysków,
- wykorzystania dostępności mniejszych, tańszych dysków klasy „PC-towej" jako alternatywy dla kupowania coraz to większych dysków najnowszej technologii.

Na przestrzeni lat technologia RAID dobrze sprawdziła się w praktyce, do tego stopnia, że niezawodne systemy buduje się dziś niemal wyłącznie na bazie macierzy RAID. Jednak wykorzystywane są w nich dziś dyski raczej wyższej klasy, podczas gdy dyski klasy „PC" stały się w międzyczasie znacznie bardziej zawodne.

W związku z tym skrót RAID jest dziś często rozwijany jako: *Redundant Array of Independent Disks*.

RAID-0

Schemat RAID-0, zwany **paskowaniem** (*striping*), polega na zapisywaniu kolejnych bloków danych na kolejnych dyskach, dzięki czemu zarówno zapis jak i późniejszy odczyt są znacznie szybsze, przy założeniu że system (procesor, magistrala i sterownik dysków) są w stanie przesyłać dane szybciej niż trwają operacje zapisu/odczytu na dyskach.

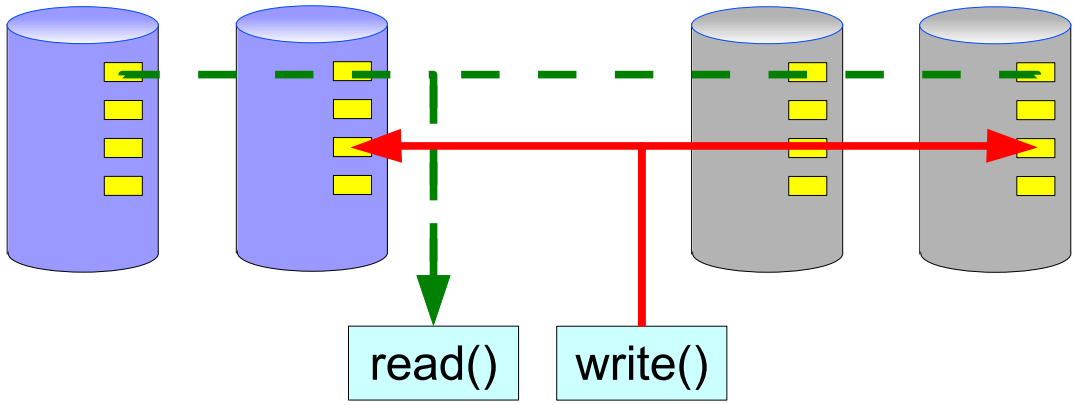


Oczywistą i bardzo poważną wadą tego rodzaju konfiguracji jest zawodność. W RAID-0 niezawodność systemu zamiast być większa, jest mniejsza niż pojedynczego dysku, ponieważ wszystkie dane zostaną utracone jeśli awarii ulegnie którykolwiek z dysków.

Z tego powodu czysta konfiguracja RAID-0 jest rzadko stosowana. Natomiast samą technikę paskowania stosuje się w innych konfiguracjach RAID w połączeniu z nadmiarowością.

RAID-1

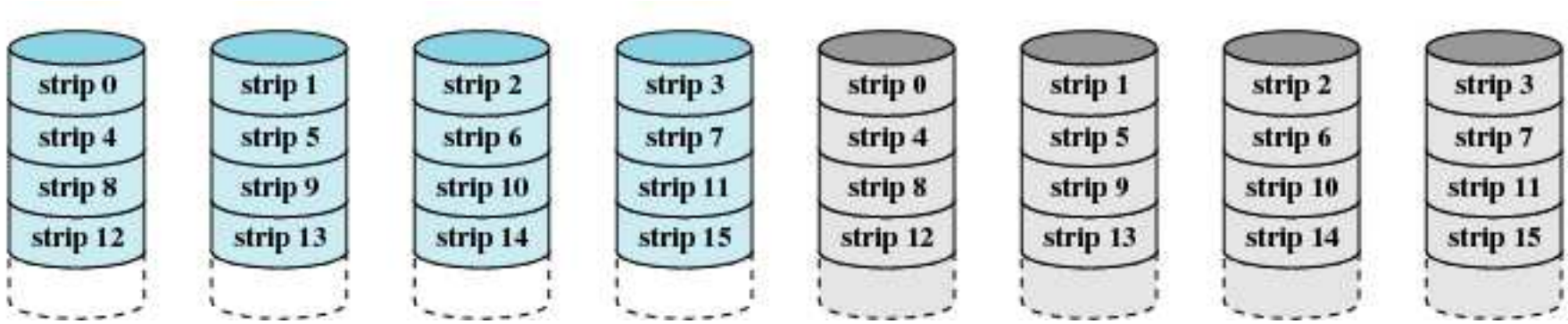
Schemat RAID-1 polega na **duplikowaniu** (*mirroring*) wszystkich operacji zapisu na dwa równoważne robocze zestawy dysków. Oczywiście operacje odczytu wystarczy wykonywać tylko z jednego zestawu. W przypadku awarii któregokolwiek z dysków jednej strony lustra system może kontynuować pracę, korzystając z drugiej kopii.



Ten niezbyt wyrafinowany schemat nadmiarowości wymaga 100% więcej dysków niż jest danych i jest najbardziej kosztownym ze wszystkich schematów RAID. Ale dzięki temu odczyt może być zrealizowany znacznie szybciej niż w przypadku pojedynczego dysku, zwłaszcza w środowisku wielodostępnym, gdzie drugi proces może jednocześnie czytać zupełnie inne dane. Jednak prędkość zapisu w schemacie RAID-1 jest ograniczona przez najwolniejszy z dysków.

RAID-10 i RAID-01

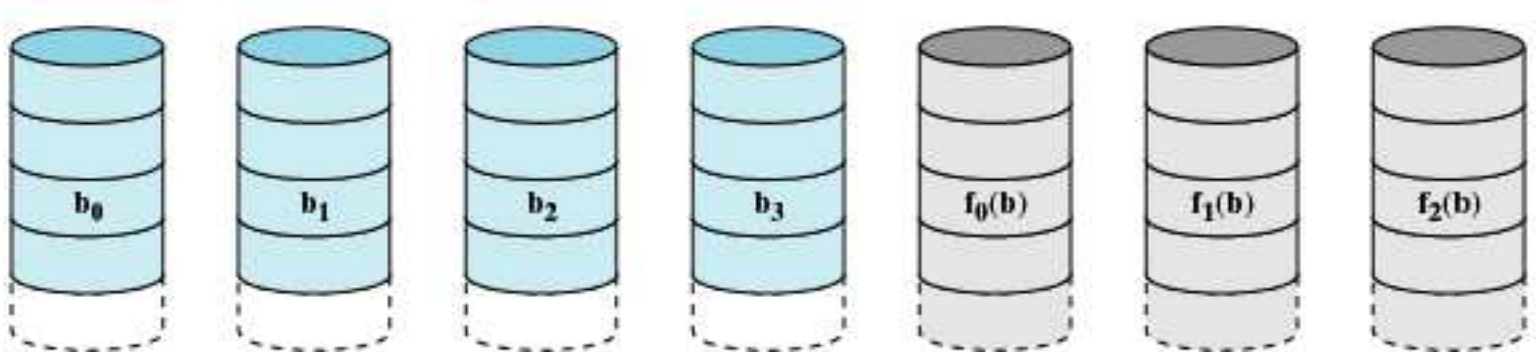
Każdy zestaw wielu dysków można używać w układzie paskowanym, zyskując na prędkości zapisu i odczytu.



Na powyższym rysunku widać połączenie struktury lustra z paskowaniem, nazywane RAID-0+1 albo RAID-01. Stosowane jest również paskowanie zestawów kopii lustrzanych, zwane RAID-1+0 albo RAID-10. W wyższych schematach RAID (4–6) paskowanie stosuje się z zasady, i nie uwzględnia się tego w nazwie.

RAID-2

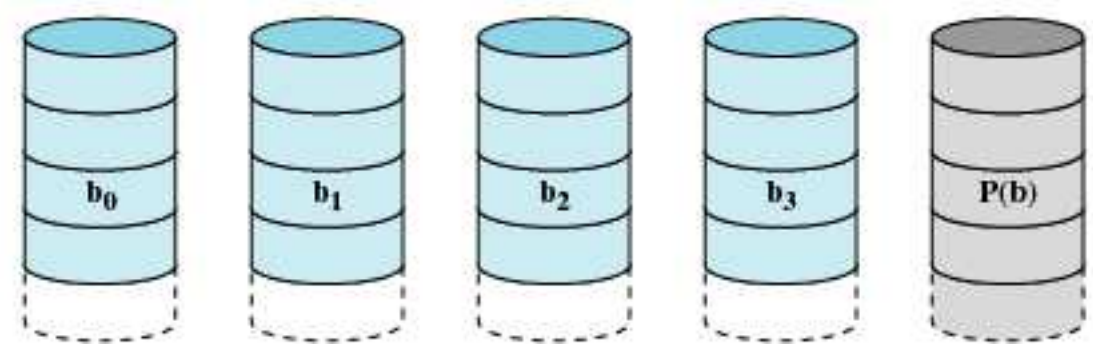
Schemat RAID-2 wykorzystuje zestaw dysków roboczych, do których dodany jest drugi zestaw dysków do kontroli i korekty błędów. Zwykle wykorzystywane są kody Hamminga pozwalające na korektę błędów 1-bitowych i wykrywanie błędów 2-bitowych. Do przechowywania kodów wymagana jest liczba dysków równa logarytmowi liczby dysków roboczych. To mniej od 100%-owego narzutu w schemacie RAID-1, ale sterownik musi obliczać bardziej złożone kody przy każdej operacji odczytu i zapisu.



Pomimo niewątpliwych zalet teoretycznych, ten schemat jest rzadko stosowany, ponieważ jest nadal kosztowny, mało elastyczny, a zapewniany poziom zabezpieczenia rzadko jest wymagany, ze względu na dużą niezawodność dysków.

RAID-3

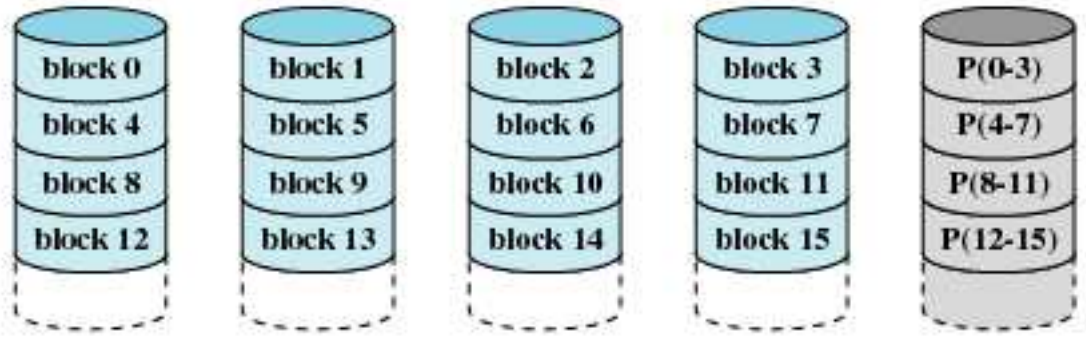
Schemat RAID-3 jest podobny do RAID-2 z tą różnicą, że zamiast kodów korygujących stosowany jest pojedynczy bit parzystości do zestawu dysków. Pozwala to na wykrywanie i korektę jednobitowych błędów, i co istotne, na kontynuowanie pracy macierzy po awarii jednego z dysków. Dane z brakującego dysku są obliczane w locie jako alternatywa wykluczająca (*exclusive-or*) pozostałych bitów z bitem parzystości.



Schemat RAID-3 ustępuje pod pewnymi względami wyższym schematom RAID. Zauważmy, że przy bitowym rozproszeniu danych każdy odczyt wymaga wspólnej (synchronicznej) pracy wszystkich dysków. Jest więc również rzadko stosowany.

RAID-4

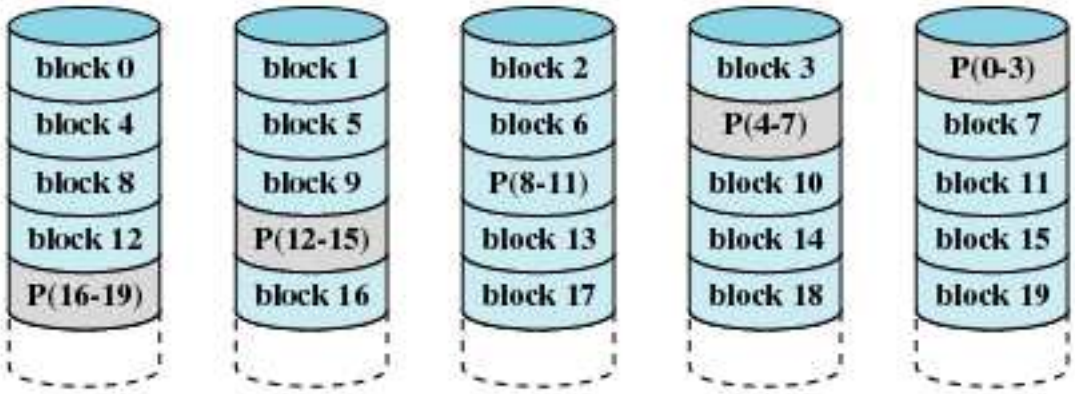
W schemacie RAID-4 zamiast rozproszenia bitów stosowane jest paskowanie bloków, i blok parzystości na dodatkowym dysku jak w RAID-3. Jednak parzystość nie jest obliczana z wektora bitów z wszystkich dysków, tylko indywidualnie dla każdego bloku, i odpowiedniej długości wektor bitów parzystości zapisywany jest w odpowiadającym sektorze na dysku parzystości. Dzięki temu odczyt każdego bloku musi być tylko zweryfikowany odczytem z dysku parzystości. Po awarii dowolnego dysku system kontynuuje pracę.



Zauważmy, że w RAID-4 obciążenie dysku parzystości jest sumą aktywności wszystkich pozostałych dysków. Jest to jedyna wada tego schematu, aczkolwiek powodująca, że w praktyce również nie jest on stosowany.

RAID-5

Schemat RAID-5 jest podobny i różni się od RAID-4 jedynie tym, że nie ma on wydzielonego dysku parzystości, a sektory zawierające parzystość są rozproszone systematycznie na wszystkich dyskach. Dzięki temu obciążenie jest rozłożone równomiernie pomiędzy wszystkimi dyskami, i schemat RAID-5 jest najczęściej stosowanym schematem macierzy RAID pojedynczej parzystości.



Podsumowując, schemat RAID-5 zapewnia:

- niezawodność dzięki parzystości; zdolność pracy po awarii dowolnego dysku,
- redundancję uzyskaną kosztem jedynie $1+n$ dysków,
- zwiększenie prędkości zapisów i odczytów dzięki paskowaniu,
- równomierną pracę i zużycie wszystkich dysków,
- ALE: przy większych zestawach dysków redundancja $1+n$ jest trochę iluzoryczna.

Redundancja jednobitowa

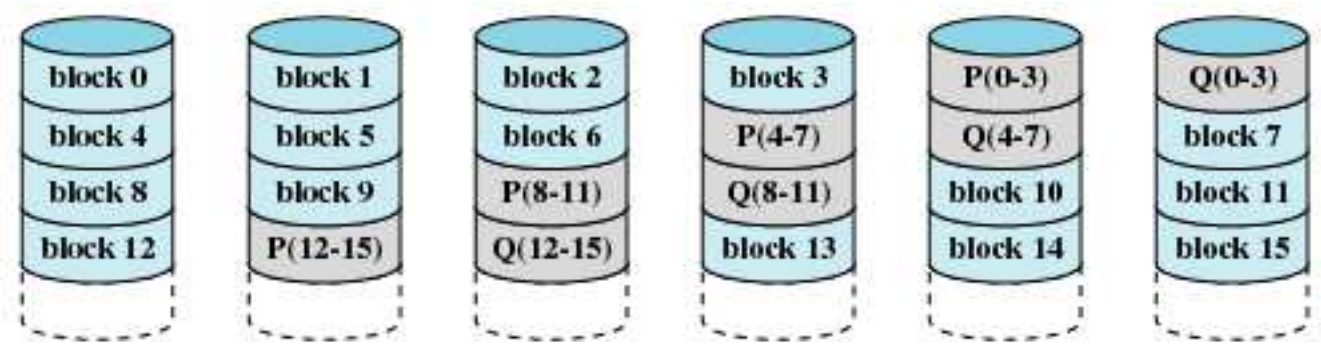
Warto podkreślić istotne cechy redundancji zapewnianej przez obliczanie parzystości, powszechnie stosowanej zamiast kodów korekcyjnych Hamminga. Oczywistą jej cechą — i główną zaletą — jest niższy koszt wynikający z zastosowania mniejszej liczby nadmiarowych dysków, oraz do pewnego stopnia, prostsze algorytmy obliczane w locie przez sterownik macierzy.

Konsekwencją jest, zamiast korekty błędów jednobitowych i wykrywania błędów dwubitowych, tylko zdolność wykrywania jednobitowych błędów oraz uzupełniania brakujących danych w przypadku awarii jednego dysku. W większości przypadków jest to pożądana i wystarczająca funkcjonalność. Jednak np. w przypadku uszkodzenia sterownika jednego z dysków powodującego błędne zapisy, ale braku przerwy w pracy, macierz wykryje błędy, ale nie będzie w stanie odtworzyć w locie poprawnych danych, bo nie będzie wiedzieć który dysk przekłamał dane. Konieczne jest zlokalizowanie wadliwego dysku i wyłączenie go z macierzy.

Kolejną cechą jest, że dla dowolnej wielkości zestawu dysków dodawany jest zawsze dokładnie jeden dysk na parzystość. Obniża to cenę i koszty eksploatacji macierzy (energia, odprowadzanie ciepła, izolacja hałasu), ale praktycznie obniża poziom bezpieczeństwa. Przy dużej liczbie dysków awarie są częste, i kluczowe jest nie samo posiadanie macierzy RAID, ale skuteczna obsługa przypadków awarii dysków.

RAID-6

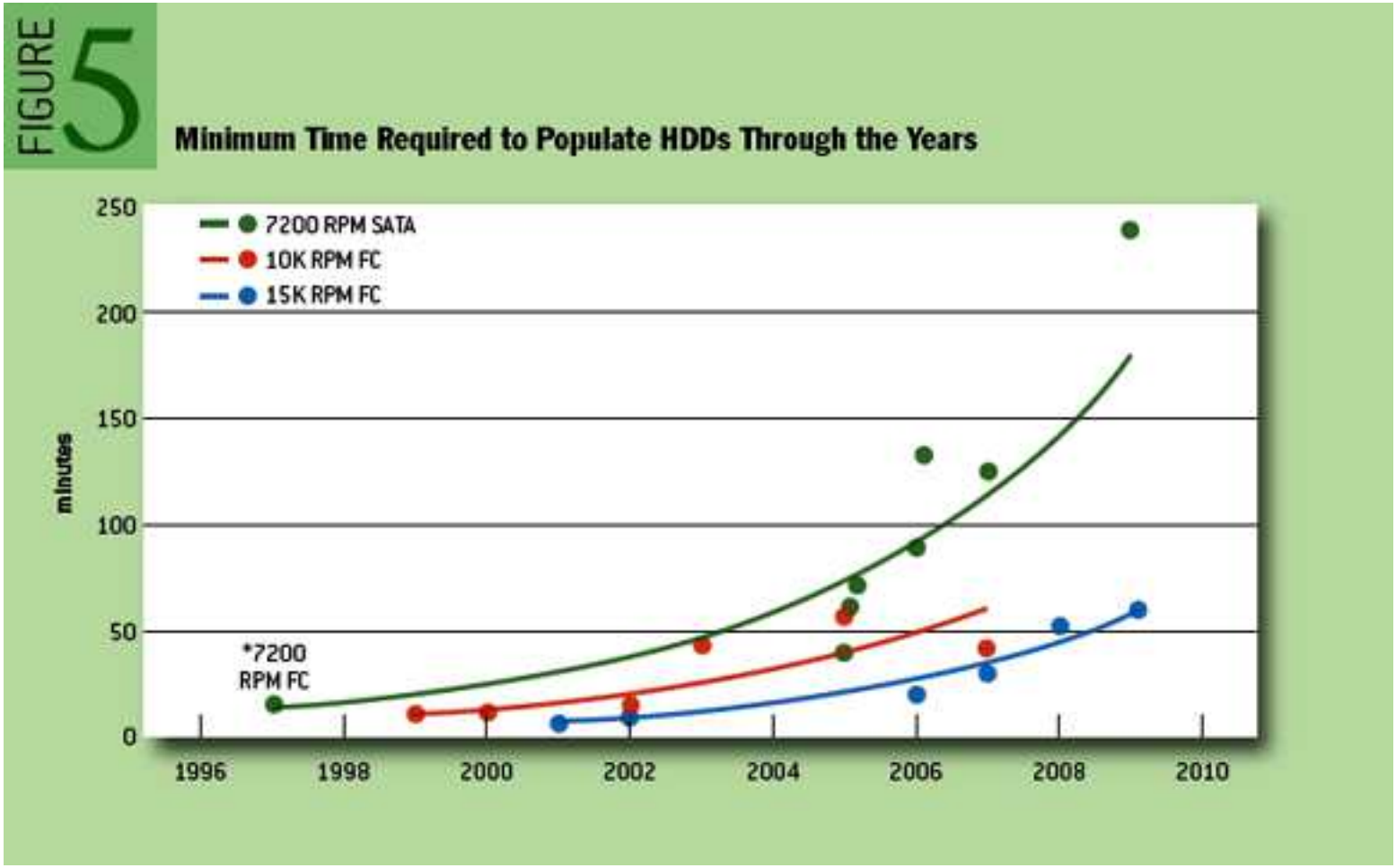
Schemat RAID-6 oparty jest na podobnej zasadzie co RAID-5, ale wykorzystuje dwa dyski parzystości obliczające sumy kontrolne różnymi metodami. Istnieją różne implementacje RAID-6 wykorzystujące różne algorytmy obliczania dodatkowej parzystości.



Macierz RAID-6 może „przeżyć" awarię dwóch dysków naraz, co ma rosnące znaczenie w macierzach o bardzo wielkiej pojemności. Przy pojedynczej parzystości, w okienku czasowym po zaistnieniu awarii dysku, do chwili jego wymiany, i pełnej odbudowy stanu, macierz pracuje praktycznie bez redundancji. Z pewnym prawdopodobieństwem w tym oknie czasowym wystąpi druga awaria, tym razem katastrofalna! Zastosowanie podwójnej parzystości, w połączeniu z szybką wymianą uszkodzonych dysków, pozwala tę możliwość praktycznie wyeliminować.

Dygresja: pojemność dysków a czas transferu

Można zauważyć, że rozwój technologii dysków powoduje niemal niezmienne podwajanie ich pojemności każdego roku, któremu jednak nie towarzyszy adekwatny wzrost przepustowości. Na skutek tego gwałtownie (szybciej niż liniowo) rośnie taki parametr jak czas niezbędny do zapisu całego dysku. Zatem wydłuża się czas odbudowy macierzy po awarii, czyli okres, w którym pozbawiona jest ona redundancji.



Przyszłość macierzy RAID

Po awarii dysku w macierzy konieczna jest jego wymiana na nowy, a następnie odtworzenie jego całej zawartości na podstawie treści pozostałych dysków. Rosnący czas zapisu całego dysku powoduje, że przez coraz dłuższy okres czasu macierz musi pracować bez pełnej redundancji. Jednocześnie zwiększona pojemność oznacza rosnące prawdopodobieństwo błędu, ponieważ niezawodność dysków również nie rośnie w tempie równym wzrostowi pojemności. W związku z tym postuluje się powstanie nowego standardu RAID potrójnej kontroli parzystości, który zapewniłby utrzymanie współczesnej niezawodności dla przyszłych, znacznie większych niż dzisiejsze, macierzy.

[1] Adam Leventhal — „Triple-Parity RAID and Beyond", December 17, 2009, `http://queue.acm.org/detail.cfm?id=1670144`.

Krótkie podsumowanie — pytania sprawdzające

1. Jakie są motywy i najważniejsze cele stosowania macierzy dyskowych RAID?

2. Na czym polega paskowanie i jakie są jego główne zalety i wady?

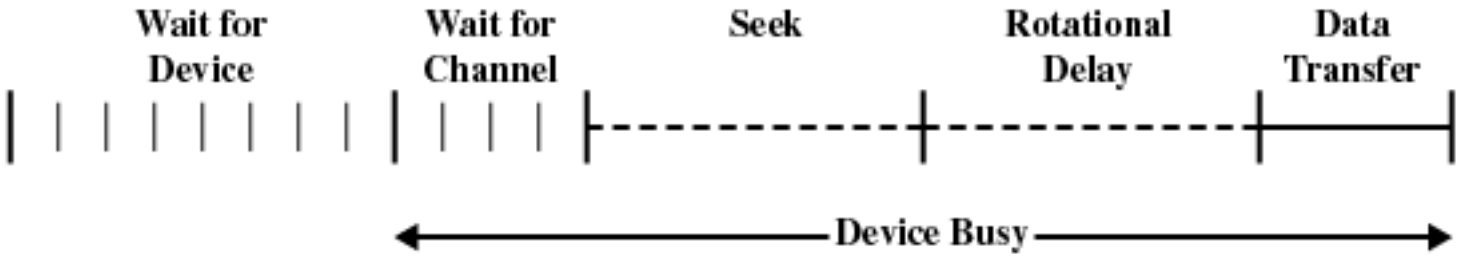
3. Na czym polega duplikowanie i jakie są jego główne zalety i wady?

4. Jaki kompromis zachodzi w doborze wielkości macierzy RAID, w której istnieje jeden dodatkowy dysk redundancji? Jakie są konsekwencje tworzenia mniejszych lub większych takich macierzy, z jednym dyskiem redundancji?

5. Opisz schemat RAID-5.

Szeregowanie operacji dyskowych

Czas wykonywania operacji dyskowej składa się z czasu wykonywania operacji przez system operacyjny realizujący żądanie, oraz czasu wykonania tej operacji przez dysk i kontroler.



Ponieważ opóźnienia i czas wykonywania operacji I/O przez system dyskowy jest na ogół znacznie dłuższy niż czas operacji procesora i/lub kontrolera dyskowego, zatem gdy w krótkim czasie wygenerowanych zostanie wiele żądań operacji dyskowych, istotne znaczenie ma kolejność, w jakiej będą one realizowane.

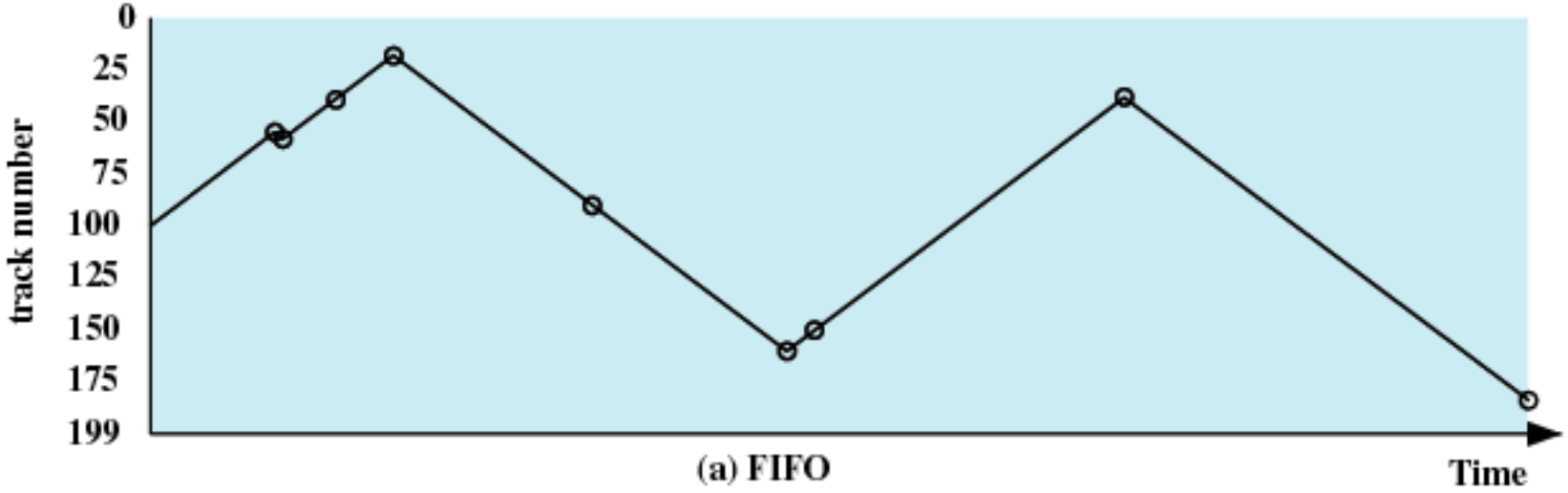
Jest to problem szeregowania operacji dyskowych.

Rozważmy przykład: dysk ma 200 ścieżek, głowica początkowo znajduje się nad ścieżką nr 100, i kolejno pojawiają się żądania dostępu do ścieżek: 55, 58, 39, 18, 90, 160, 150, 38, 184. Jeśli pojawiły się one w krótkim czasie, system ma możliwość ustawienia kolejności ich realizacji.

Strategia FCFS

First Come First Serve: żądania obsługiwane są w kolejności nadejścia.

Przykład: dysk ma 200 ścieżek, głowica początkowo znajduje się nad ścieżką nr 100, i kolejno pojawiają się żądania dostępu do ścieżek: 55, 58, 39, 18, 90, 160, 150, 38, 184.

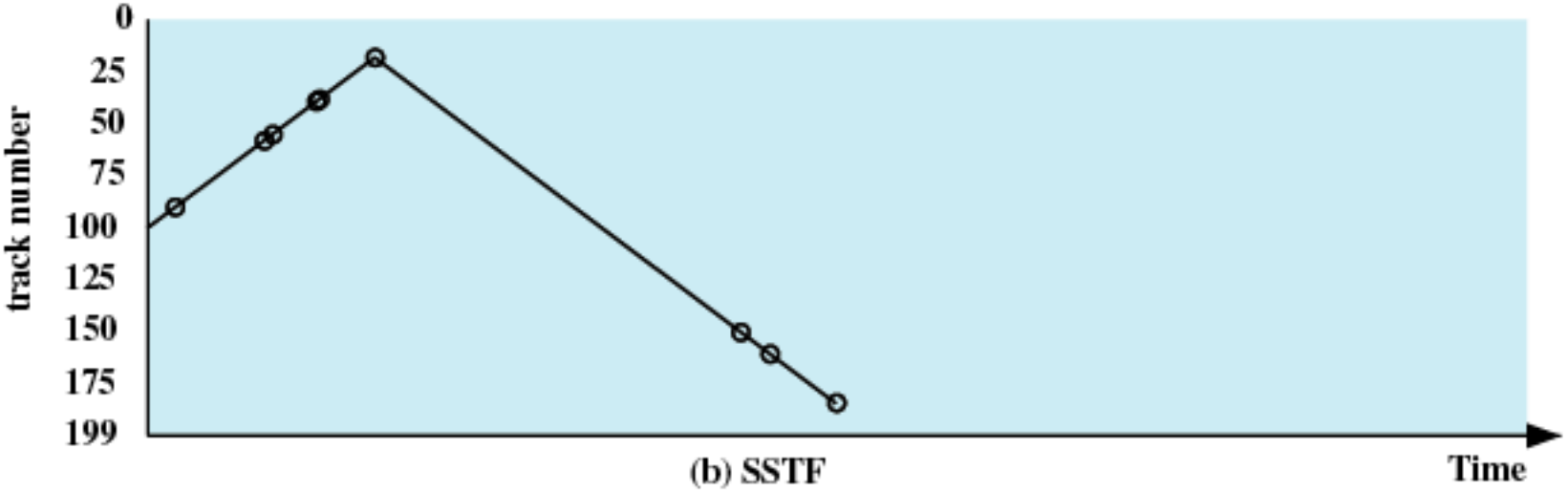


Zauważmy, że do realizacji strategii FCFS potrzebna jest tylko znajomość czasów napływania zadań. Można ją realizować nawet gdy system szeregujący nie zna dokładnego stanu urządzenia dyskowego.

Strategia SSTF

Shortest Service Time First: najpierw obsługiwane jest żądanie o najmniejszym czasie seek (przesunięcia głowicy).

Przykład: dysk ma 200 ścieżek, głowica początkowo znajduje się nad ścieżką nr 100, i kolejno pojawiają się żądania dostępu do ścieżek: 55, 58, 39, 18, 90, 160, 150, 38, 184.

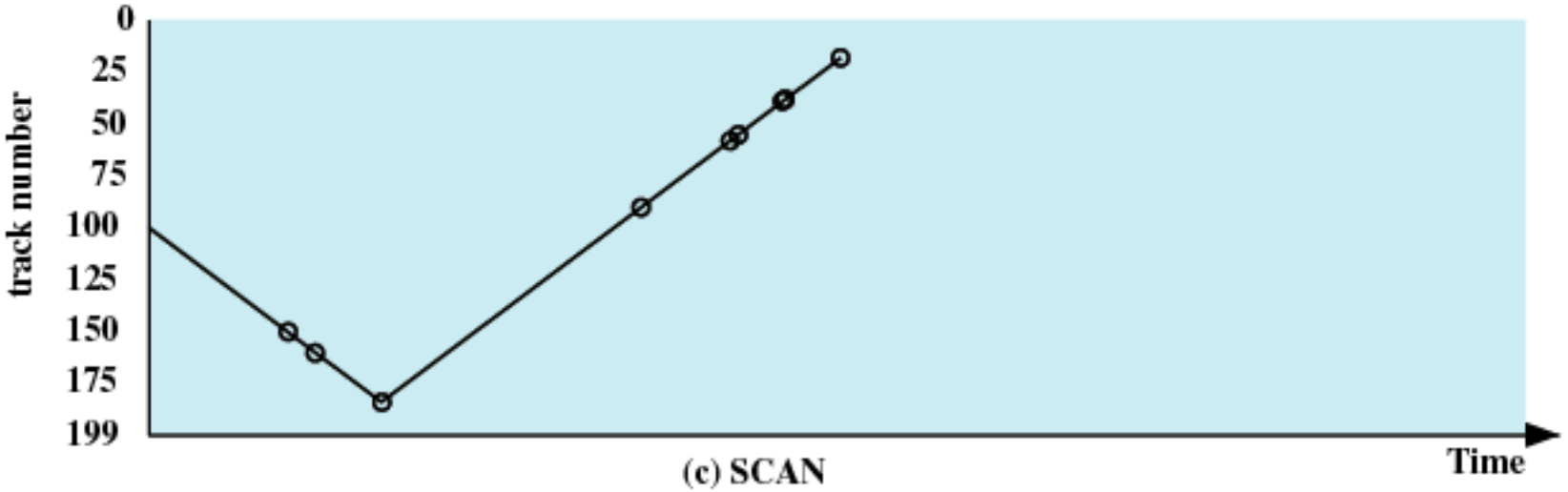


Zauważmy, że strategia SSTF jest w pewnym sensie optymalna (widać to nawet na powyższym przykładzie). Jednak może ona w pewnym okresie doprowadzić do zmonopolizowania systemu dyskowego, jeśli jeden proces będzie wysyłał nieprzerwanie żądania dotyczące ścieżek w jednej okolicy.

Strategia SCAN

SCAN (algorytm windy): głowica przesuwa się nad całą powierzchnią talerza w jedną stronę i obsługuje wszystkie żądania dotyczące mijanych cylindrów.

Przykład: dysk ma 200 ścieżek, głowica początkowo znajduje się nad ścieżką nr 100, i kolejno pojawiają się żądania dostępu do ścieżek: 55, 58, 39, 18, 90, 160, 150, 38, 184.

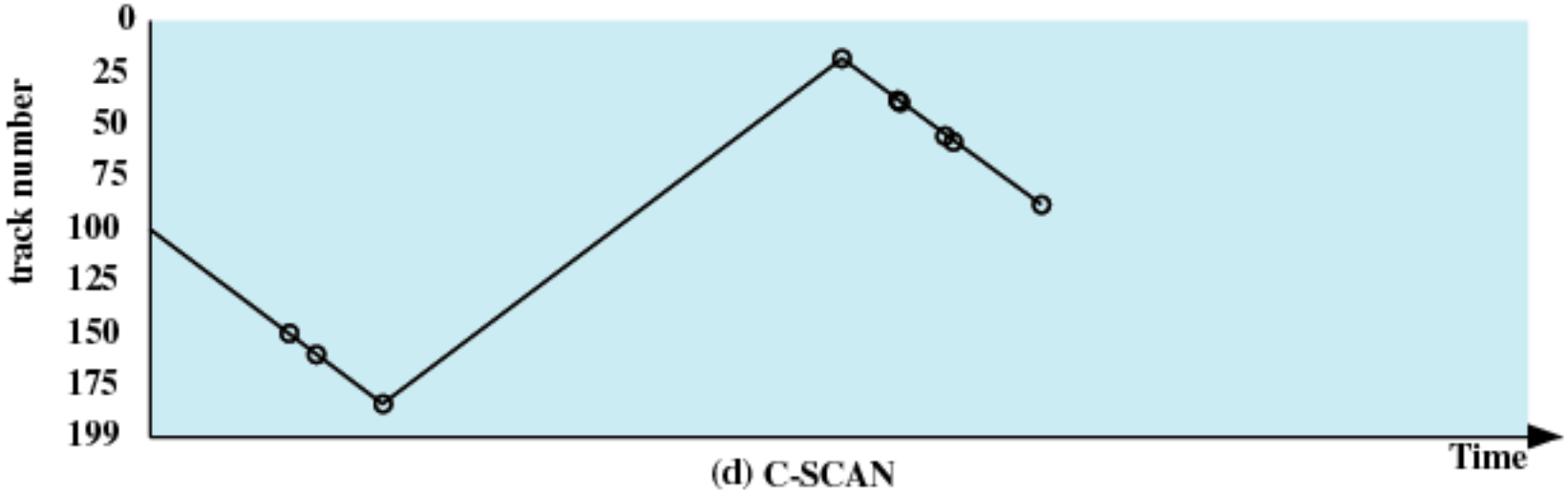


Strategia SCAN eliminuje (w pewnym stopniu) możliwość zagłodzenia systemu przy strategii SSTF. Oprócz podstawowej wersji (głowica przesuwa się od pierwszej do ostatniej ścieżki, i z powrotem), stosuje się jej wersję zwaną LOOK, gdzie głowica zawraca gdy nie ma już dalszych zgłoszeń w kierunku dotychczasowego przesuwania.

Strategia C-SCAN

C-SCAN (*circular scan*): działa jak SCAN podczas przesuwania głowicy w jedną stronę, po czym wraca na początek dysku bez obsługiwania żadnych żądań i zaczyna od początku.

Przykład: dysk ma 200 ścieżek, głowica początkowo znajduje się nad ścieżką nr 100, i kolejno pojawiają się żądania dostępu do ścieżek: 55, 58, 39, 18, 90, 160, 150, 38, 184.



C-SCAN (a szczególnie wariant C-LOOK) zachowuje zasadnicze cechy strategii SCAN, a jednocześnie zapobiega faworyzowaniu nowych zgłoszeń kosztem tych najdłużej oczekujących w kolejce. Skraca ona maksymalny czas oczekiwania na serwis (być może kosztem średniego czasu).

Inne strategie

W przypadku strategii SSTF, ale także SCAN, i C-SCAN pewne nadchodzące żądania mogą na dłuższy czas zmonopolizować system dyskowy. Można zastosować zmodyfikowane wersje strategii SCAN aby temu zapobiec.

FSCAN — istnieją dwie kolejki żądań obsługiwane na zmianę. W czasie obsługiwania jednej z nich, nadchodzące żądania są dodawane do drugiej.

N-step-SCAN — nadchodzące żądania są umieszczane w podkolejkach o długości maksymalnie N. Żądania nadchodzące w czasie realizacji jednej z podkolejek są umieszczane w innej. Jeśli w czasie realizacji jednej kolejki nadejdzie nie więcej niż N zgłoszeń, to wszystkie będą obsłużone w czasie kolejnego cyklu.

Zauważmy, że wybór wartości N w efekcie steruje strategią obsługi. Przy dużej wartości N zachowanie algorytmu N-step-SCAN jest podobne do SCAN. DIa N=1 realizacja N-step-SCAN sprowadza się do strategii FCFS.

Porównanie strategii szeregowania

(a) FIFO (starting at track 100)		(b) SSTF (starting at track 100)		(c) SCAN (starting at track 100, in the direction of increasing track number)		(d) C-SCAN (starting at track 100, in the direction of increasing track number)	
Next track accessed	**Number of tracks traversed**	**Next track accessed**	**Number of tracks traversed**	**Next track accessed**	**Number of tracks traversed**	**Next track accessed**	**Number of tracks traversed**
55	45	90	10	150	50	150	50
58	3	58	32	160	10	160	10
39	19	55	3	184	24	184	24
18	21	39	16	90	94	18	166
90	72	38	1	58	32	38	20
160	70	18	20	55	3	39	1
150	10	150	132	39	16	55	16
38	112	160	10	38	1	58	3
184	146	184	24	18	20	90	32
Average seek length	55.3	**Average seek length**	27.5	**Average seek length**	27.8	**Average seek length**	35.8

Krótkie podsumowanie — pytania sprawdzające

1. Jakie są fazy realizacji operacji dyskowej, i co wpływa na czas wykonywania poszczególnych faz?

2. Wymień z nazwy i zdefiniuj co najmniej trzy znane Ci strategie szeregowania operacji dyskowych.

3. Jaki parametr usiłują optymalizować strategie szeregowania operacji dyskowych, który algorytm najlepiej optymalizuje ten parametr, i jakie są wady tego algorytmu?

4. Opisz jedną wybraną strategię szeregowania operacji dyskowych, którą uważasz za dobrą, i wskaż jej zalety w porównaniu z innymi.