NTNU
Institutt for datateknikk
og informasjonsvitenskap

Eksamensoppgave i

TDT4225 Lagring og behandling av store datamengder

Lørdag 18. mai 2013, kl. 0900-1300

Oppgaven er utarbeidet av faglærer Kjell Bratbergsengen og kvalitetssikrer Svein-Olaf Hvasshovd

Kontaktperson under eksamen: Kjell Bratbergsengen, telefon 7359 3439 og 906 17 185

Språkform: Bokmål

Tillatte hjelpemidler: D
Ingen trykte eller håndskrevne hjelpemiddel er tillatt.
Bestemt, enkel kalkulator tillatt.

Sensur innen:tirsdag 11. juni 2013.

Oppgave 1, Sekvensoptimalisering av I/O (15%)

For å effektivisere I/O kan diskkontroller bruke forskjellig strategi for å betjene I/O-ordrene.
a) Forklar strategiene FCFS first come first served, SSTF-shortest seek time first og SCAN.

b) Følgende operasjoner skal utføres på en disk med 1000 sylindre, bare sylinder-delen av adressen er gitt. Lesehodets starposisjon er sylinder 500. Hva blir total posisjoneringsdistanse for de forskjellige strategiene beskrevet i opp a) når adressene er:
903, 396, 442, 112, 222, 741, 832.
Anta at SSTF alltid har tre forespørsler i køen (unntatt for de siste to operajonene). SCAN samler opp alle forespørslene før den starter behandlingen.

Forslag til løsning

a) Se i læreboka.

b)

500	FCFS	SSTF		SCAN	
903	403	58	442	388	112
396	507	46	396	110	222
442	46	174	222	174	396
112	330	110	112	46	442
222	110	629	741	299	741
741	519	91	832	91	832
832	91	71	903	71	903
	2006	1179		1179	

Alle besøkte adresser er i venstre kolonne. For FCFS er det også betjeningsrekkefølgen. Antall sylindre overflyttet står i kolonne 2.
Kolonne 2 under SSTF angir spor nummer som er besøkt. Kolonne 1 er antall spor krysset for å komme fram. Tilsvarende for SCAN. SCAN går først til laveste sylinder i køen og betjener alle på vei mot høyeste spor i køen. Andre varianter av SCAN kan benyttes.
FCFS tvinger oss til å posisjonere over totalt 2006 spor. SSTF og SCAN gir samme resultat 1179 spor. Posisjoneringsarbeidet er redusert til nesten det halve ved å bruke en strategi. SCAN reduserer lengste ventetid i forhold til SSTF.

Oppgave 2, 20 % Videoserver

En videoserver skal levere film til individuelle brukere som ser på forskjellige filmer. Hver film krever en tilnærmet konstant datastrøm over tid på 3 MB/s. Lokalt har hver skjerm et lokalt lager som holder data for 15 sekunder film. Filmene ligger på en SSD (solid state disk). Disken er koplet til en systembussen med en optisk kabel som kan overføre opp til 300 MB/s. Systembussen har en kapasitet på 800 MB/s. Anta at maskinen s arbeidslager kan ”holde følge ” med systembussens kapasitet. Arbeidslager er 1 GB. Dataoverføringer styres av en DMA-enhet.

a) Vis hvordan data flyter over de forskjellige systemdelene.

b) SSD har et tidsforbruk ved lesing som følger formelen t(x)=a+bx. x er blokkstørrelse i antall byte. a er 0,1 millisekund og b regnes ut på bakgrunn av at lesehastigheten er 200 MB/s. Hvilken systemdel (disk, optisk kabel, systembuss) er flaskehals når ingen av disse skal utnyttes til mer enn 70% av toppkapasitet.

c) Hvor mange samtidige skjermer kan serveren betjene? Hvilken blokkstørrelse har du valgt å bruke i overføringene.

Forslag til løsning

a) Data går fra SSD over optisk fiberkabel til en I/O adresse på bussen. DMA flytter data fra I/O-port til buffer i arbeidslager. Fra arbeidslager flytter DMA data til utport for transmisjon til videostasjon. Hvis DMA mellomlagrer data i seg selv skal data over bussen 4 ganger. Hvis DMA kan operere i fly-by –modus flyttes data direkte mellom I/O-port og arbeidslager. Da halveres bussbelastningen

b)

	MB/s	0,7			
Optisk kabel	300	210			70,0
Systembuss	800	140			46,7
Disk	200	0,0001	100% kap	70% kap	
blokk	2048	0,00011	18,6	13,0	4,3
	4096	0,00012	34,0	23,8	7,9
	8192	0,000141	58,1	40,7	13,6
	16384	0,000182	90,1	63,0	21,0
	32768	0,000264	124,2	86,9	29,0
	65536	0,000428	153,2	107,3	35,8
	131072	0,000755	173,5	121,5	40,5
	262144	0,001411	185,8	130,1	43,4
	524288	0,002721	192,7	134,9	45,0
	1048576	0,005343	196,3	137,4	45,8
	2097152	0,010586	198,1	138,7	46,2
	4194304	0,021072	199,1	139,3	46,4

Nederste del av denne tabellen viser I/O-kapasitet som funksjon av blokkstørrelse.
Grenseverdien for SSD er 140 MB/s det samme som for systembussen når DMA gjør mellomlagring. Store blokker 0,5 -4 MB gir høy kapasitet. Det er SSD som blir flaskehals.

c) 45 skjermer kan betjenes med blokker på 0,5 MB.
Vi trenger 2 buffre , dvs. 1 MB per skjerm. Totalt 45 MB. Vi kan bruke flere bufre enn to per datastrøm eller vi kan bruke større bufre. Med 2 MB bufre klarer vi en skjerm til. Det er det også god plass til.

Oppgave 3, (10%) LRU og Clock
a) Forklar LRU algoritmen ? Hvorfor er algoritmen dårlig ved lesing av en sekvensiell fil?

b) Forklar Clock-algoritmen.

c) Vi har et arbeidslager med plass for 4 bufre. Gitt følgende referansestreng for objekter:
a,b,c,d,b,d,e,a,b,b,d,c,e
i) Hvor mange blokker må leses når LRU-algoritmen brukes.
ii) Hvor mange blokker må leses når Clock-algoritmen brukes.

Forslag til løsning

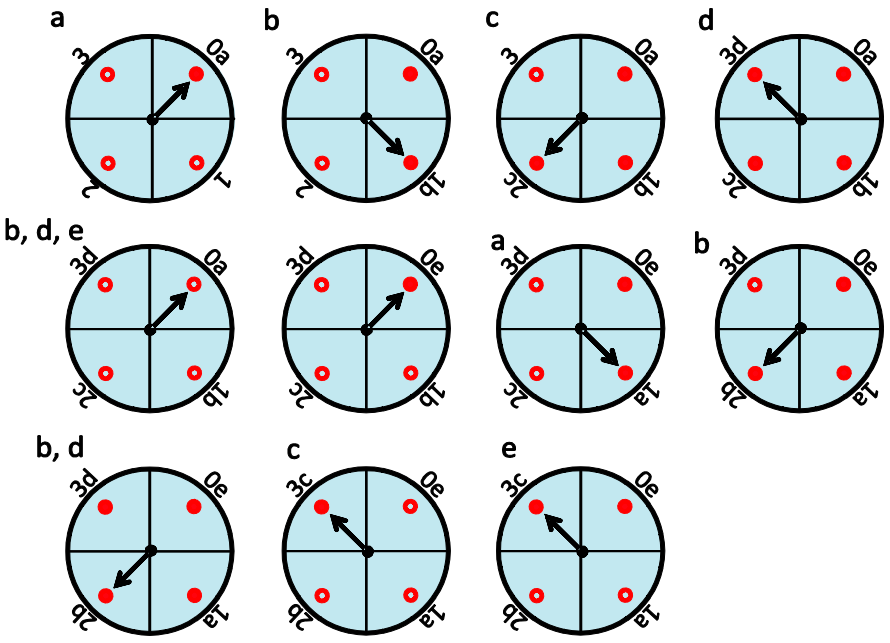
a) LRU gir den sist brukte blokk topp prioritet. For sekvensiell lesing er det ugunstig, fordi sist leste blokk skal ikke leses igjen (innen rimelig tid).

b) Søker lineært gjennom bufrene, sirkulært. Hvis blokken (buffer) er brukt siden siste gang søkeren var på besøk vil objektet ha et fredet-merke. Fredet-merket settes til ikke fredet. Hvis søkeren kommer til en ikke-fredet buffer blir den tatt.

c) LRU fører til at 8 blokker må leses inn fra sekundærlager.

	LRU												
Priority	a	b	c	d	b	d	e	a	b	b	d	c	e
1	a	b	c	d	b	d	e	a	b	b	d	c	e
2		a	b	c	d	b	d	e	a	a	b	d	c
3			a	b	c	c	b	d	e	e	a	b	d
4				a	a	a	c	b	d	d	e	a	b
5							a	c	c			e	a

Blokker med prioritet 5 eller lavere har mistet sin buffer. Gule blokker på prioritet 1 må leses inn i buffer.



Clock gir også 8 innleste blokker. Clock er vist i figuren. En fylt sirkel representerer en buffer som er referert siden siste viser-passering.Viseren står på den bufferen som sist ble tatt.
Øverste rekke: a b c og d er referert og neste buffer er ledig.
Midterste rekke: b d e er referert. e er uten buffer, alle bufre er referert siden siste viserpassering, derfor må viseren gjøre en runde før den treffer på en buffer som kan tas. Deretter refereres a og b som begge trenger bufre.
Nederste rekke: b refereres. d trenger buffer, alle er referert og viser gjør en full runde før den finner buffer for d. c trenger buffer, e har buffer og bekrefter referanse.

Vi kan også sette opp en alternativ figur:

	CLOCK														
Buffer	a	b	c	d	b	d	e		a	b	b	d	c		e
0	a	a	a	a	a	a	a	e	e	e	e	e	e	e	e
1		b	b	b	b	b	b	b	a	a	a	a	a	a	a
2			c	c	c	c	c	c	c	b	b	b	b	b	b
3				d	d	d	d	d	d	d	d	d	d	c	c

Viseren står på rød buffer. Gule og røde buffere er markert som referert til siden siste viserpassering. Referansene e og c har dobbel bredde. Der må viseren gjennom alle bufre for å finne en ledig.

Oppgave 4, (20%) Lagring av rasterbilder.

Et stort bilde er lagret i blokker, hver blokk inneholder et bildekvadrat. Hvert pixel i bildet tar 4 byte.

a) Hvis bildet lagres i blokker som hver tar 32 KB (32768 byte) hvor stor blir da sidekanten i bildekvadratet som en blokk inneholder?

b) Fra bildet tas det utsnitt, vilkårlig plassert, men med sidekanter som er parallelle med bildekantene. Størrelsen på utsnittet er 2000 x 2000 piksler. Hvor stort blir I/O-volumet for uthenting av slike utsnitt i gjennomsnitt? Blokkstørrelsen er fortsatt 32 KB.

c) Bildet lagres på disk som roterer 15000 ganger per minutt. Hvert spor lagrer 0,5 MB. Hva blir overføringstiden for å hente ett utsnitt som beskrevet i punkt b)

Forslag til løsning

a) En blokk inneholder 8192 pixler, sidekanten blir 90 pixler, dvs. blokken inneholder 8100 pixler.

b) Volumet som hentes ut er V=w(h+x)(w+x)=4x2090*2090=17472400=17,4 MB

c)

$$T = \frac{V}{\omega}\left(\frac{1}{b} + \frac{1}{V_s}\right)$$

Block size	32768	65536	131072	262144	524288	1048576	2097152	4194304
rpm	15000	15000	15000	15000	15000	15000	15000	15000
Vs	524288	524288	524288	524288	524288	524288	524288	524288
w	4	4	4	4	4	4	4	4
h	2000	2000	2000	2000	2000	2000	2000	2000
b	2000	2000	2000	2000	2000	2000	2000	2000
x	90	128	181	256	362	512	724	1024
omega	250	250	250	250	250	250	250	250
Volum	17472400	18113536	19027044	20358144	22316176	25240576	29680704	36578304
T	2,266	1,244	0,726	0,466	0,341	0,289	0,283	0,314

Tabellen viser tidsforbruket for en serie blokkstørrelser. For 32 KB er T=2,266 sekunder.
Tabellen viser også T for forskjellige blokkstørrelser. En blokkstørrelse på ca. 2 MB er nær optimal (ikke spurt om til eksamen).

Oppgave 5, UNIX (15%)

a) Forklar datastrukturen som brukes av UNIX filsystem for å komme fra logisk intern adresse innen en fil til en fysisk adresse på disk.

b) Et program skriver en string på 1000 byte til fil på følgende adresser:
0, 5000, 10000, 15000, 20000.
Hvor mange blokker blir allokert til filen. Installasjonen bruker en blokkstørrelse på 4KB (4096 byte).

Forslag til løsning

a) se læreboka.

b)

UNIX filssystem			
0	0	999	4095
4096	5000	5999	8191
8192	10000	10999	12287
12288	15000	15999	16383
16384	20000	20999	20479
20480	20000	20999	

Alle adresser er innenfor 10 blokker og ingen ekstra indeksblokk blir allokert.

Oppgave 6, Relasjonsalgebra (20%)

	Resultat	Basetabeller			Mellomresultat		
	R	A	B	C	A*B=B*A	A*C=C*A	B*C=C*B
Nøkkellengde i byte	8	8	8	8	8	8	8
Postlengde i byte	200	1200	400	700	200	200	200
Antall poster	300 000	500 000	1 200 000	50 000	400 000	200 000	100 000
Tabellvolum i MB	60	600	480	35	80	40	20

En skal utføre operasjonen **R=A*B*C.** Informasjon i tabellen over. Nøkkellengde gjelder join-operasjonens nøkkel.

a) Finn den sekvens av operasjoner som gir minst I/O-volum. Anta at mellomresultater er lagret på fil. Ikke vurder join-algoritmer eller ta hensyn til eventuelle interne mellomlagringer i operasjonene i denne deloppgaven.

b) Finn den beste algoritmen for å gjøre operasjonen **U=A*B.** Resultatpostene i **U** henter 100 byte fra hver av operandene. Tilgjengelig arbeidslager WS er 10 MB.

Forslag til løsning

a)

Antall mulig sekvenser av operasjoner og operander er 12. Volummessig er operandsekvens symmetrisk slik det er oppgitt her. Mulige sekvenser er: (AB)C A(BC), (AC)B A(CB). Alle operander må leses inn en gang uansett sekvens. For å fullføre må en forene et mellomresultat med en full tabell. Vi velger den sekvens som gir minst mellomresultat. Det er BC eller CB.

A	600
B	480
C	35
R	60
BC*2	40
Total	1215

b) Join A*B. WS er 10 MB. Aktuelle størrelser:

	Resultat	Basetabeller			Mellomresultat		
	R	A	B	C	A*B	A*C	B*C
Nøkkellengde i byte	8	8	8	8	8	8	8
Postlengde i byte	200	1200	400	700	200	200	200
Antall poster	300 000	500 000	1 200 000	50 000	400 000	200 000	100 000
Tabellvolum i MB	60	600	480	35	80	40	20
Resultatdel		100	100		200		
Nettofil med data		50	120		80		
Bare nøkkeldata		4	9,6		3,2		

Vi skal forene A med B. Vi prøver da flere metoder for å finne den som gir minst samlet transportvolum.

Gjentatte gjennomløp.

WS er 10 MB. Vi skal hente 100 byte fra hver av operandene til resultatpost. Antall poster i A er 0,5 M og i B 1,2 M. Totalt volum som skal stilles opp i WS er 50MB, antall gjennomløp blir da 5. Siden B-postene er store i forhold til 100 byte fra B-post som skal inn i U-postene, er det lønnsomt å skrive en temporærfil for B-postene: TB.

Volum blir 600+480+5*120+80=**1760 MB**.

Nested Loop	
Les A	600
Les B	480
Skriv TB	120
Les TB 4 ganger	480
Skriv A*B	80
I/O-volum i MB	1760

Partisjonering

Partisjonering gir 600+480+(50+120)*2+80=**1500 MB**

Partisjonering	
Les A	600
Skriv TA splitt	50
Les B	480
Skriv TB splitt	120
Les TA og TB	170
Skriv A*B	80
I/O-volum i MB	1500

Nøkkelfiltrering

Det er plass til alle A-nøkler i WS. Det kan vi bruke til å filtrere bort alle B-poster som ikke kan inngå i resultatet når vi splitter B.

Les alle A-poster og still opp nøkkel i WS, skriv også TA til fil, splitt i for eksempel 8 partisjoner
Les alle B-poster, alle med tilslag i WS skrives til fil TB, som splittes etter samme mønster som TA.
Les til slutt TA og TB partisjon for partisjon og fullfør join-operasjonen.

Antall resultatposter i A*B er 400000, det er også det maksimale antall poster i TB. Trolig er det noe mindre, men vi regner 400000 for å bruke øvre grense. Det er en tredjedel av alle postene i B.

Partisjonering og nøkkelfilter	
Les A	600
Skriv TA splitt	50
Les B	480
Skriv TB splitt 1/3	40
Les TA og TB	90
Skriv A*B	80
I/O-volum i MB	1340

Bruk av nøkkelfiltrering gir best resultat, **1340 MB**.

Kjb/25.4.2013