Bufferhåndtering i multimedia datahåndteringssystemer

Pål Halvorsen
Universitetsstudiene på Kjeller (UniK)

i: *Elektronikk, årg. 32, nr. 9, 1998, s. 64-67*

Sammendrag

Kontinuerlig avspilling av store, komplekse og tidsavhengige multimedia data som video krever at datahåndteringssystemene effektivt kan hente ut data fra disk til buffer. Her presenteres Q-L/MRP som er en bufferhåndteringsmekanisme spesielt designet for interaktive, tidsavhengige datastrømmer med støtte for tjenestekvalitet.

1 Innledning

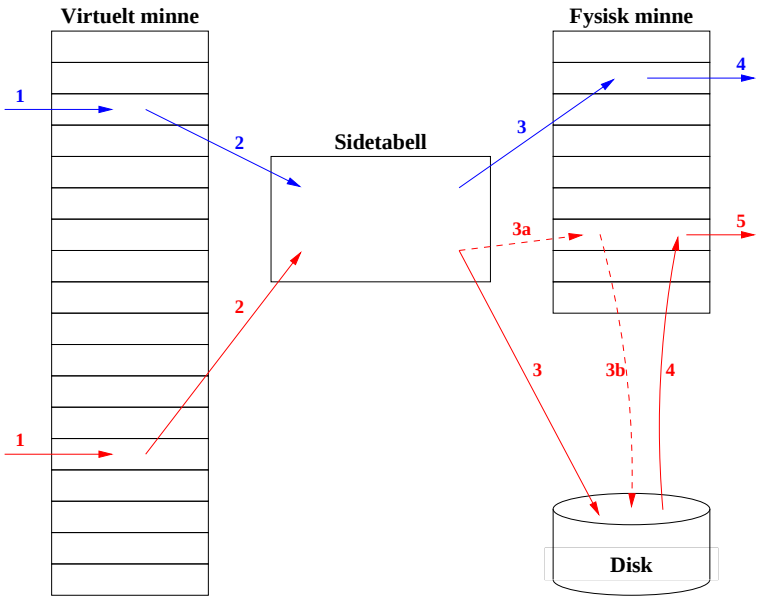
Distribuerte multimedia applikasjoner, som nyheter-på-forespørsel (News-on-Demand), digitale biblioteker og asynkron, interaktiv fjernundervisning, ser ut til å bli en viktig del av fremtidens informasjonssamfunn. Slike applikasjoner må effektivt kunne håndtere store, komplekse, kontinuerlige og tidsavhengige datatyper som audio og video. Et viktig begrep i denne konteksten er tjenestekvalitet (Quality-of-Service, QoS). Tjenestekvalitet beskriver kravene fra brukerene/applikasjonene på den ene siden og ytelsen til og påliteligheten av multimediasystemet på den andre. Spesifikasjon av tjenestekvalitet gir muligheten til å effektivt håndtere systemressursene, som nettverk, disk, minne og prosessorer, og å gjøre reservasjoner for å sikre den ønskede kvaliteten på avspillingen av multimedia data.

2 Interaktiv fjernundervisning

I DEDICATION, et prosjekt for databasestøtte i fjernundervisning på UniK, ser vi på utfordringene med å støtte flere, samtidige avspillinger av lagrede forelesninger fra et såkalt elektronisk klasserom. I dag kan de elektroniske klasserommene vi har blant annet i Oslo og på Kjeller, tilby studenter å følge en forelesning på forskjellige geografisk adskilte klasserom. Dette tilbys ved direkte overføring av video, audio og data fra et "whiteboard" mellom klasserommene (synkron fjernundervisning). Målet er imidlertid å kunne overkomme adskillelse både i tid og rom mellom foreleser og student. I DEDICATION ønsker vi å løse dette ved å lagre forelesninger i et multimedia databasesystem, slik at vi i fremtiden også skal kunne støtte adskillelse i tid. Det vil si at en student kan få avspilt video og audio fra en forelesning når studenten selv vil (asynkron fjernundervisning).

3 Bufferhåndtering

I den lange kjeden av systemkomponenter som effektivt må kunne håndtere multimediadata og støtte tjenestekvalitet, er håndteringen av bufferet (minnet) en viktig del. Figur 1 viser et forenklet, skjematisk bilde over bufferet i en datamaskin. En forespørsel om data i det virtuelle minnet blir oversatt i en sidetabell for å se om de ønskede dataene ligger i det fysiske minnet eller på disken. Som figuren viser er det to muligheter. Det optimale (vist i blått i figuren) er om dataene finnes direkte i det fysiske minnet. Her kan vi bare returnere adressen til dataene som så blir prosessert. Problemet er bare at bufferet er en begrenset ressurs, og grunnet den begrensede størrelsen kan ikke alle dataene



Figur 1: Et forenklet, skjematisk bilde over minnet i en datamaskin. De blå pilene viser en forespørsel hvor dataelementet allerede finnes i minnet. De røde pilene viser en forespørsel hvor dataelementet først må hentes fra disken til det fysiske minnet.

ligge i bufferet til en hver tid. Vi må derfor ofte hente data fra disk til buffer (vist i rødt i figuren). I dette tilfellet må vi først finne data i bufferet som kan byttes ut, og hvis disse dataene er modifisert må disse skrives tilbake til disken. Deretter leses de ønskede data fra disk til riktig plass i bufferet og adressen til bufferelementet med de nye dataene returneres.

En viktig punkt for optimal bruk av bufferet (og dermed også ytelsen til hele systemet) er hvordan data hentes inn og byttes ut i bufferet. Disker er per idag altfor langsomme, og overføring av data fra disk til buffer tar lang tid. For å best mulig kunne støtte mange samtidige brukere i vår asynkrone, interaktive fjernundervisnings-applikasjon og for å i det hele tatt kunne tilby en kontinuerlig fremvisning av video, må antall diskaksesser minimeres ved å best mulig tilpasse bufferhåndteringsmekanismen. Videre er dagens disker for langsomme til at vi kan hente et og et dataelement i det øyeblikket dataene trengs (demand paging). Vi trenger en form for forhåndshenting av data slik at antall diskaksesser går ned og den totale tiden det tar å overføre data fra disk til buffer reduseres [4].

4 Eksisterende algoritmer

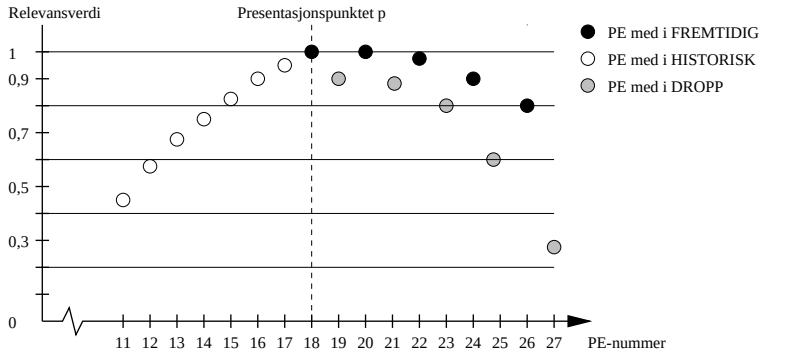
For å finne en passende bufferhåndteringsmekanisme for vår interaktive fjernundervisningsapplikasjon, har vi undersøkt om det allerede finnes velegnede mekanismer. Våre analyser viser at tradisjonelle mekanismer som FIFO, LRU og RANDOM [2] ikke er brukbare. Heller ikke deres forbedringer, LRU-K [8] og 2Q [5], er egnet for multimedia applikasjoner. Problemet med disse mekanismene er at de ikke er designet for multimedia data. Tradisjonell bufferhåndtering benytter bare dataelementenes alder (hvor lenge de har vært i bufferet) og referansehyppighet (hvor mye har dataene blitt brukt) som kriterier for å bytte ut data i bufferet med data fra disken. De tar ikke i betraktning den karakteristiske referansestrukturen til multimedia data, og vi risikerer at disse algoritmene bytter ut de dataene vi trenger neste gang. Videre henter disse mekanismene et og et dataelement fra disk til buffer. For å kunne presentere data fra tidsavhengige data som video, må vi forbedre dataoverføringen fra disk med å forhåndshente større mengder med data for hver dataforespørsel.

I tillegg til de tradisjonelle bufferhåndteringsmekanismene, er det designet og implementert en mengde nye mekanismer, for eksempel [1], [6] eller [7], som er utviklet for multimedia data. Pro-

blemet med disse er at de bare støtter homogene tjenestekvalitetskrav. Det vil si at en bruker ikke selv kan spesifisere hvordan dataene skal presenteres. Videre trenger de veldig store buffere for å fungere. Vi har i DEDICATION en veldig begrenset bufferkapasitet og kan ikke lagre alle data mellom to samtidige brukere. Sist, men ikke minst, støtter de ikke interaktive brukere. De tilbyr ingen utvidet funksjonalitet for brukere som vil hoppe litt frem og tilbake i fremvisningen. I vår interaktive fjernundervisningsapplikasjon er heterogene tjenestekvalitetskrav og støtte for en høy grad av interaktivitet to kjernepunkter, og så langt har vi ikke funnet noen eksisterende mekanismer som støtter dette.

5 Q-L/MRP

I stedet for å designe en helt ny bufferhåndteringsmekanisme til vår interaktive fjernundervisningsapplikasjon, har vi valgt å utvide L/MRP (Least/Most Relevant for Presentation) [7] utviklet ved GMD Darmstadt. Dette er en bufferhåndteringsmekanisme spesielt utviklet for interaktive datastrømmer som vi har tilpasset våre krav. I den utvidede mekanismen, kalt Q-L/MRP (QoS-L/MRP) [4], har vi lagt til støtte for multiple, samtidige datastrømmer (flere brukere og medietyper) og støtte for tjenestekvalitet. Vi har omgjort forhåndshentingsmekanimen fra å hente et dataelement av gangen til en dynamisk, adaptiv forhåndshentingsdemon. Denne reduserer antall diskaksesser ved å senke frekvensen på diskaksessene og ved å hente større mengder data hver gang. Mengden av data og forhåndshentingsfrekvensen avhenger av og justeres dynamisk etter systemets arbeidsmengde slik at de ønskede data kan forhåndshentes til bufferet i tide uavhengig av forsinkelser og trafikk i nettverk og I/O-system.



Figur 2: Beregning av relevansverdier og inndeling i interaksjonssett i en hurtig avspilling fremover [7]: x-aksen viser hvilket nummer de forskjellige PEene, y-aksen viser relevansverdiene og den stiplede linjen viser hvilken PE som presenteres (presentasjonspunktet).

For å sikre at vi ikke bytter ut data vi trenger neste gang, byttes data ut etter hvilke data som er minst relevant for presentasjonen, se figur 2. Først deler vi dataelementene (presentasjonsenhetene, PEene) inn i såkalte interaksjonssett avhengig av hvilken presentasjonsmodus vi har. En PE kan ha blitt presentert, skal presenteres eller skal droppes (for eksempel ved hurtigspoling med bilde). Avhengig av hvilket interaksjonssett PEen tilhører, beregner vi en relevans verdi som viser hvor sannsynlig det er at denne PEen kommer til å bli brukt med den presentasjonsmodusen vi har nå. Som figuren viser, vil PEer nærmest presentasjonspunktet ha høyest relevansverdi, og relevansverdien synker gradvis avhengig av interaksjonssettet PEen tilhører og avstanden til presentasjonspunktet. Når så alle relevansverdier er beregnet, byttes PEer med minst relevansverdi ut.

6 Resultater

For å undersøke om Q-L/MRP er egnet til vår interaktive fjernundervisningsapplikasjon, har vi implementert og simulert Q-L/MRP i matlab. Som simuleringsdata har vi brukt et vilkårlig valgt fem minutters vindu fra en lagret forelesning, og vi har simulert både interaktive og lineære avspillinger. Vi har variert side- og bufferstørrelser, antall brukere og nettverksbelastninger. I tillegg har vi simulert Q-L/MRP i både klient og server.

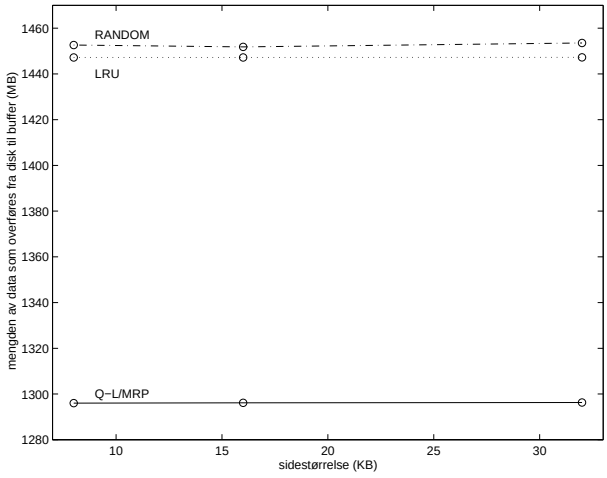
Vår første observasjon fra simuleringsresultatene er at Q-L/MRP er i stand til å kontinuerlig (uten avbrudd) presentere multimedia data fra vår lagrede forlesning. Det betyr at antall sidefeil (forespørsel om data som ikke finnes i bufferet og dermed må hentes fra disk) er redusert til et minimum. På klientsiden gir Q-L/MRP "perfekte" presentasjoner av forelesningene hvor resultatene viser feilfrie lineære avspillinger og en sidefeil i de interaktive avspillingen. På serversiden reduseres også antall sidefeil til et minimum. Lineære avspillinger gir opp til to sidefeil, mens interaktive presentasjoner gir opp til 174 sidefeil. I scenarioet med 3 brukere, en bufferstørrelse på 64MB og 16KB sider, gir Q-L/MRP cirka 0,09% sidefeil i forhold til LRU.

Simuleringsresultatene viser også at Q-L/MRP reduserer dataoverføringene fra serverens disk til brukerens buffer ved å bruke en bedre tilpasset uttbyttingsalgoritme og ved å ta hensyn til brukerenes tjenestekvalitetsspesifikasjoner:

- Ved å bytte ut data som er minst relevant for presentasjonen risikerer vi ikke å bytte ut data som må hentes inn igjen i nærmeste fremtid. Data som brukes flere ganger hentes inn bare en gang og den totale dataoverføringen reduseres. Et eksempel er gitt i figur 3: i et scenario med tre brukere og et buffer på 64MB, gir Q-L/MRP en kontinuerlig presentasjon av den lagrede forelesningen, men overfører cirka 10.3% mindre data enn LRU.

- Hvis en bruker bare ønsker halve rammeraten (for eksempel hvis ikke modemet kan ta i mot data raskere), er det bortkastede ressurser å hente inn alle rammene. I dette tilfellet vil Q-L/MRP bare hente inn annen hver ramme slik at bare halvparten av ressursene brukes i forhold til algoritmer a la LRU.

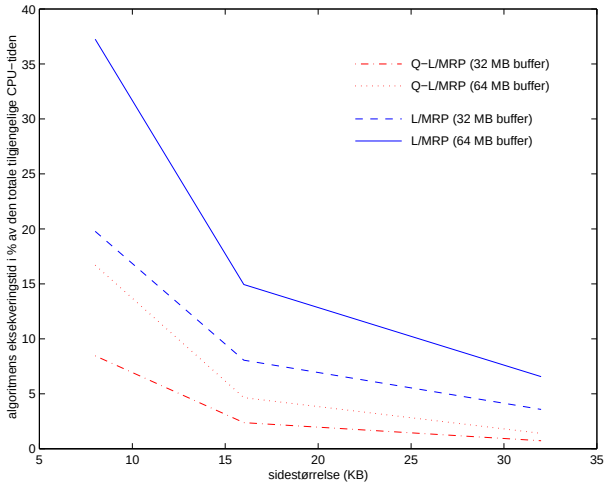
En slik forbedret ressursbruk kan igjen brukes til å håndtere flere samtidige brukere eller brukes til andre formål.



Figur 3: Sammenligning av mengden data overført fra disk til buffer ved bruk av Q-L/MRP, LRU og RANDOM. Figuren viser et scenario med tre brukere, et buffer på 64MB og en lineær avspilling.

Vårt siste evalueringspunkt var eksekveringstiden av selve koden til Q-L/MRP. Simuleringsresultatene fra figur 4 viser at det er en forholdsvis kostbar algoritme å eksekvere og at eksekveringstiden

øker med antall brukere, større buffere og mindre sidestørrelser. Ved bruk av et buffer på 32MB vil den totale eksekveringstiden med sidestørrelser på 16KB og 32KB være henholdsvis på 2,37% og 0.73% av den totale tilgjengelige CPU-tiden. Likevel, som figur 4 viser, er eksekveringstidene sterkt redusert i forhold til L/MRP.



Figur 4: Sammenligning av eksekveringstider ved bruk av L/MRP og Q-L/MRP i et en-bruker scenario. Figuren viser også økende eksekveringstider ved større buffere og mindre sidestørrelser.

7 Konklusjon

Vi har i denne artikkelen sett at bufferhåndteringen er en potensiell flaskehals i et multimedia datahåndteringssystem. Analyser av eksisterende mekanismer viser at det enda ikke er utviklet og implementert noen algoritmer som fullt ut støtter heterogene tjenestekvalitetskrav og en høy grad av interaktivitet i en kontinuerlig presentasjon av multimedia datastrømmer. Basert på kravene fra DEDICATION-prosjektet, har vi designet Q-L/MRP som er en bufferhåndteringsmekanisme spesielt egnet for interaktive, tidsavhengige datastrømmer. Q-L/MRP er en videreutvikling av L/MRP med støtte for multiple datastrømmer og støtte for tjenestekvalitet ved en dynamisk adaptiv forhåndshentingsdemon.

Vår ytelsesanalyse viser at Q-L/MRP støtter kontinuerlig presentasjon av multiple, tidsavhengige datastrømmer (både samtidige brukere og forskjellige medietyper), og mengden av data overført fra disk til buffer er sterkt redusert. Likevel er Q-L/MRP en kompleks algoritme hvor eksekveringstiden øker med antall brukere, større bufferstørrelser og mindre sidestørrelser. Konklusjonen er derfor at Q-L/MRP er overlegen andre eksisterende bufferhåndteringsmekanismer med hensyn til interaktivitet og støtte for tjenestekvalitet. Q-L/MRP er derfor meget godt egnet i en klient, men grunnet algoritmens kompleksitet og høye eksekveringstider, kan den bli for kostbar i en server.

En direkte fortsettelse av dette arbeidet vil være en analyse av et system med Q-L/MRP på klientsiden og en annen mekanisme som for eksempel “media caching” [1], [6] på serversiden. Generelt er vårt arbeid her på UniK knyttet til støtte av tjenestekvalitet i alle systemkomponentene i en asynkron, interaktiv fjernundervisningsapplikasjon hvor vi blant annet i OMODIS-prosjektet har integrasjon av tjenestekvalitet i et multimedia databasesystem som hovedformål [3].

Referanser

[1] Dan, A., Sitaram, D.: “Multimedia Caching Strategies for Heterogeneous Application and Server Environments”, Multimedia Tools and Applications, Vol. 4, No. 3, May 1997, pp. 279 - 312

[2] Effelsberg, W., Haerder, T.: “Principles of Database Buffer Management”, ACM Transactions on Database Systems, Vol. 9, No. 4, December 1984, pp. 560 - 595

[3] Goebel, V., Plagemann, T., Berre, A.-J., Nygård, M.:“OMODIS - Object-Oriented Modeling and Database Support for Distributed Systems”, Norsk Informatikk Konferanse (NIK’96), Alta, Norway, November 1996, pp. 7 - 18

[4] Halvorsen, P., Goebel, V., Plagemann, T.: “Q-L/MRP: A Buffer Management Mechanism for QoS Support in a Multimedia DBMS”, Proceedings of 1998 IEEE International Workshop on Multimedia Database Management Systems (IW-MMDBMS’98), Dayton, Ohio, USA, August, 1998, pp. 162 - 171

[5] Johnson, T., Shasha, D.: “2Q: A Low Overhead High Performance Buffer Management Replacement algorithm”, Proceedings of the 20th VLDB Conference, Santiago, Chile, 1994, pp. 439 - 450

[6] Kamath, M., Ramamritham, K., Towsley, D.: “Continuous Media Sharing in Multimedia Database Systems”, 4th Int. Conf. on Database Systems for Advanced Applications (DASFAA’95), Singapore, April 1995, pp. 79 - 86

[7] Moser, F., Kraiss, A., Klas, W.: “L/MRP: A Buffer Management Strategy for Interactive Continuous Data Flows in a Multimedia DBMS”, Proceedings of the 21th VLDB Conference, Zurich, Switzerland, 1995, pp. 275 - 286

[8] O’Neil, E., J., O’Neil, P., E., Weikum, G.: “The LRU-K Page Replacement Algorithm For Database Disk Buffering”, Proceedings of the 1993 ACM SIGMOD International Conference on Management of Data, Washington, D.C., USA, May 1993, pp. 297 - 306