Institutionen för Informatik
Handelshögskolan vid Göteborg Universitet

Magisteruppsats IA7400, 20p

Animationsdesign för algoritmvisualisering

Jens Corsman och Håkan Wessberg

9 juni 2000

Handledare:

Kjell Engberg, Insitutionen för Informatik, Göteborgs Universitet
Philippas Tsigas, Institutionen för Datalogi, CTH/GU

Sammanfattning
Att använda bilder och visualiseringar för att förklara koncept och händelseförlopp är något som har gjorts länge, även inom algoritmundervisning. Det ställs dock speciella krav på animering av algoritmer då dessa animationer måste konkretisera något väldigt abstrakt samtidigt som visualiseringen måste vara generell. Forskning om animationer som undervisningshjälpmedel för algoritmundervisning har dock inte kunnat påvisa några positiva effekter av att använda animationer. I denna uppsats har vi försökt besvara frågan om hur man skall designa visualiseringar för att dessa skall stödja inlärning. För att besvara denna fråga har vi undersökt hur inlärningsprocessen fungerar och hur multimedia kan stödja denna. Detta har vi gjort genom att undersöka vilka resultat som har framkommit i den pedagogiska forskningen om multimediala undervisningshjälpmedel och de psykologiska teorier som ligger till grund för denna forskning för att se om dessa resultat är applicerbara på design av algoritm animationer. Övrig forskning som vi anser kan ge designimplikationer har också studerats. Den pedagogiska forskningen av Richard Mayer som bygger på psykologiska koncept introducerade av Allan Paivio visade sig vara direkt tillämpbar på design av algoritmanimationer och även forskning om färg som informationsförmedlare visade sig nyttig.
Utifrån dessa teorier formulerar vi sedan ett antal kriterier som en visualisering måste uppfylla för att stödja inlärning. För att demonstrera hur dessa kriterier kan tillämpas designar vi en prototyp för visualisering av en distribuerad algoritm samt utvärderar en redan existerande animation och jämför med den nya visualiseringen utifrån kriterierna.
Efter detta diskuterar vi ett antal punkter som vi anser vara viktiga ur design synvinkel för att en visualisering skall uppfylla kriterierna och stödja inlärning.

1 Förord

Vi vill tacka PHILIPPAS TSIGAS, institutionen för Datalogi CTH/GU, för att han har tagit sig tid att diskutera distribuerade system och animationer av sådana.
Vi vill även tacka KJELL ENGBERG för synpunkter och vägledning.
Ett stort tack till ERIK WINLÖF för hjälp med bilder till prototypkapitlet.

Innehåll

Figurer

2 Introduktion

If we knew what it was we were doing,
it would not be called research, would it?
Albert Einstein (1879-1955)

Att använda bilder och visualiseringar för att förklara koncept och händelseförlopp är något som har gjorts länge, även inom algoritmundervisning. Det känns ofta självklart att det blir lättare att förstå ett koncept ifall en animation som beskriver konceptet presenteras. Dock har forskningen om animationer som undervisningshjälpmedel för algoritmundervisning visat dåliga resultat . Det har skrivits mycket om hur man tekniskt skall implementera animationerna [6], hur interaktivitet påverkar inlärningen [4, 8] och hur studenter använder animationer [21, 22].

Enligt vår vetskap har dock ingen utrett hur visualiseringar av algoritmer bäst skall designas för att stödja inlärning. Dock har bedrivits en stor mängd allmän pedagogisk forskning som försökt förstå hur multimedia kan designas för att hjälpa inlärningprocessen.

2.1 Syfte

Syftet med denna uppsats är att undersöka hur visualiseringar skall designas för att stödja inlärning.

Avgränsning Vi kommer i den här uppsatsen främst koncentrera oss på den visuella delen av multimediala undervisningshjälpmedel. Den audiella delen kommer nämnas men inte detaljstuderas.

Målgrupp Den här uppsatsen riktar sig till lärare och forskare på universitet och högskola, med undervisning i algoritmer och datastrukturer.

2.2 Bakgrund

Traditionellt används bilder i undervisning både som komplement till verbala förklaringar och som självständiga undervisningsmaterial för att klargöra komplicerade koncept [8]. I undervisning har bilder eller andra visualiseringar använts för att visa på sammanhang mellan objekt och skeenden, abstrakta eller konkreta.

Animationer i undervisning Animationer har använts i undervisning för att förklara koncept som innehåller element vilka är svåra att visualisera i statiska miljöer. I Newtons mekanik finns till exempel ett behov av att visa kausala samband, något som i läroböcker ofta visas med pilar som verkar på olika objekt och sekvenser av bilder som visar förlopp.

Algoritmanimation När algoritmer visualiseras inträder ett dilemma, studenten saknar ofta egna mentala representationer av de koncept som algoritmen innehåller. Det ställer krav på att visualiseringen både skall konkretisera abstrakta element i algoritmen, samtidigt som visualiseringen skall presentera algoritmen tillräckligt generellt för att inte låsa representationen [41].

2.2.1 Lydian

Lydian[1] utvecklas på institutionen för Datalogi vid Chalmers Tekniska Högskola och syftar till att vara ett undervisningshjälpmedel som skall användas i undervisning av distribuerade algoritmer och protokoll. Lydian har databaser med beskrivningar av nätverk, algoritmer och animationer. Studenten kan välja att se på en animation från animationsdatabasen, eller simulera en egen eller fördefinierad algoritm. Algoritmen kan simuleras i ett existerande nätverk eller i ett som studenten ritar själv [24].

Gemensamma grafiska element Tidigt i utvecklingen av Lydian bestämdes att alla animationer skulle använda samma grafiska objekt för att presentera information om de objekt som återfinns i många animationer, till exempel noder och meddelanden. Meddelanden mellan enheter i det distribuerade system åskådliggörs alltid på samma sätt i visualiseringarna och noder representeras av samma grafiska element.

Gemensamt för alla visualiseringar är att studenten kan välja att se ytterligare information om systemets tillstånd och meddelanden som då presenteras i fyra separata fönster. De fyra fönstrena visar en specifik process tillstånd och senaste händelse, totalt antal skickade meddelande per process, meddelandekomplexitet samt processernas tillstånd per process. Dessa fyra fönster ser alltid ut på samma sätt oavsett algoritm och nätverk.

2.3 Relaterad forskning

Det har genomförts ett antal studier om visualisering och animationer som hjälpmedel vid utlärning av algoritmer och resultatet har varit genomgående negativa. Forskningen har inte lyckats visa på några direkta fördelar med animationer. Intresset för animationer som en del av algoritmundervisning väcktes av *Sorting Out Sorting* [5], en trettiominuters film som beskrev nio sorteringsalgoritmer.

Byrne, Stasko och Catrambone [8] genomförde två experiment med universitetsstudenter för att undersöka påståenden om att animationer hjälper studenter att effektivt lära sig algoritmer. De betraktade i första hand relationen mellan animationer och studenternas förmåga att göra förutsägelser om algoritmen. Studenterna fick lära sig nya algoritmer, en del av studenterna fick se animationer och en del av dessa fick uppmaningar om att förutsäga hur en algoritm skulle bete sig med viss data att operera på. Det fanns också en kontroll grupp som endast hade tillgång till skriven text och bilder. I studien var tiden studenterna tillbringade med materialet kontrollerad så att alla studenter hade ungefär lika lång tid att använda det utdelade materialet på. De kunda dock inte visa på några fördelar från att använda animationer.

Kehoe och Stasko [21] genomförde 1996 en etnografisk studie för att få inblick i vilken sorts information studenter försöker utvinna ur olika läromedia. De ville också ta reda på ifall det valda mediet innehöll den sökta infor-

[1] http://www.cs.chalmers.se/~lydian

mationen och när i en läroprocess animationer kunde vara till hjälp. Studien undersökte studenter som under 35 minuter svarade på frågor om binomial heaps. Under denna tid hade studenterna full tillgång till text, animationer, bilder och pseudo-kod. Det medium som studenterna använde mest var texten. Animationerna användes när studenterna ville förstå vilka steg som utfördes under en operation och de använde då oftast text eller pseudokod parallellt med animeringen.

Användning av animeringar som inlärningshjälpmedel har också studerats utifrån ett bredare perspektiv än enbart animeringar av algoritmer. Bland annat genomförde Elkerton och Palmiter [13] en studie där de jämförde hur animationer, text och animationer ihop med en berättare påverkade inlärning av användning av ett användargränssnitt. De förväntade sig, baserat på tidigare resultat, att de som använde sig av animationer ihop med en berättare skulle uppvisa bäst resultat då animationerna skulle underlätta den första inlärningen och berättaren skulle få dem att komma ihåg det de hade lärt sig. De kunde dock inte visa på någon sådan skillnad, båda grupperna som hade animationer som inlärningshjälp hade problem med att behålla och generalisera kunskapen. De förklarade detta med att berättad text behandlas annorlunda jämfört med skriven text eller att användarna inte var tillräckligt uppmärksamma på berättaren för att helt och hållet behandla vad som sades.

Corbett, Pane och John [10] gjorde 1996 försök med att använda multimedia för att lära ut tidsvarierande biologiska processer. De jämförde studenter som fick tillgång till datorbaserade filmer, simulationer, text och grafik med studenter som endast hade tillgång till texter och stillbilder. Trots väldefinierade mål för vad animationerna skulle bidra med till inlärningen och passande instruktionsdata kunde de inte påvisa någon fördel för de studenter med tillgång till dynamiska animationer i jämförelse med de studenter som enbart hade tillgång till text och stillbilder.

2.3.1 Visualisering av parallella program

Visualiseringen av parallella program och algoritmer är en komplicerad uppgift. För att få en konsistent bild av systemets globala tillstånd i ett givet ögonblick måste flera processer stoppas, meddelanden som överförs mellan processer kan vara svåra att få tillgång till, det kan vara skillnad i överföringshastighet mellan olika kommunikationskanaler i systemet. Slutligen, systemklockor hos olika processer kan vara osynkroniserade och dessa klockor kan dra sig med olika hastighet. Exekvering av distribuerade algoritmer kan vara ickedeterministiskt, vilket innebär att två olika körningar med samma program kan ge olika resultat och vägen till resultaten kan ha varit olika [23, 24, 42].

Eileen Kraemer vid Washington University har undersökt möjligheten att visualisera exekveringen av distribuerade program som stöd för programmerare, i form av debuggerverktyg och verktyg för att mäta prestanda [23]. Dessa verktyg riktar sig direkt till personer som själva designar algoritmer eller program av distribuerad karaktär. De exempelsystem hon tar upp kräver eget arbete av den som vill visualisera algoritmer, men ofta finns det ett antal visualiseringar som fungerar med olika algoritmer och system.

De visualiseringar Kraemer har studerat skiljer sig från det vi vill göra i det att dem är avsedda som arbetsverktyg och inte som undervisningshjälpmedel, vilket gör att hennes resultat inte är riktigt applicerbara på denna uppsats.

2.4 Disposition

Vi har disponerat denna uppsats på följande sätt:

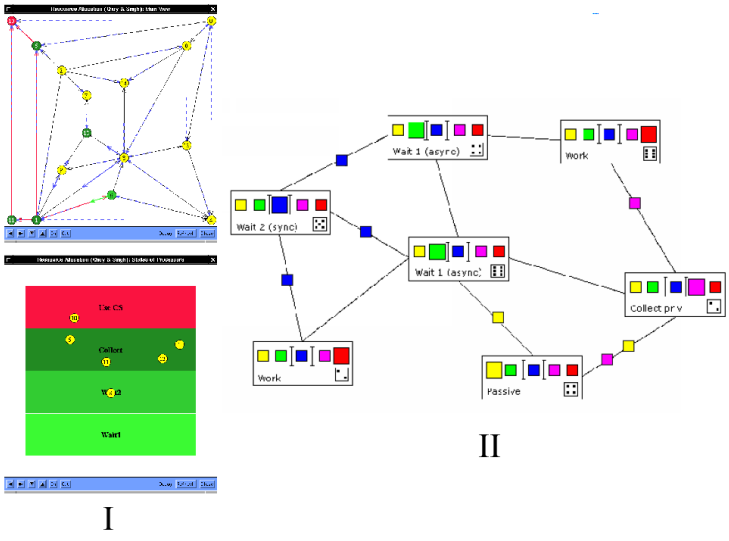
Metod Metodkapitlet innehåller en beskrivning av hur vi har gått tillväga. Det innehåller även en kort introduktion till distribuerade system då vi har använt oss av en algoritm i ett sådant för att designa en visualiseringsprototyp.

Teori Då vi skall undersöka hur visualiseringar skall designas för att stödja inlärning studerar vi i det här kapitlet teorier om hur inlärningsprocessen går till från psykologin samt forskning om hur multimedia kan stödja denna process. Även viss annan forskning som vi anser vara intressant för detta syfte, forskning om färger som informationsförmedlare och forskning om algoritmanimering, presenteras.

Utifrån dessa teorier genereras ett antal krav som måste uppfyllas för att en visuliasering skall lyckas med att stödja inlärning.

Prototyp I detta kapitel börjar vi med att presentera den algoritm som vi skall visulisera. Då delar av denna algoritm redan finns visualiserad i Lydian börjar vi med att utifrån de krav som tidigare genererats utvärdera denna algoritm. Vi designar sedan en egen animation av samma del av algoritmen och jämför den utifrån kraven med den tidigare animationen. Efter detta designas de övriga delarna av visualiseringen.

Diskussion och slutsatser Här presenterar vi de slutsatser som vi har kommit fram till och diskuterar vilka implikationer dessa får på design av algoritm visualiseringar.



Figur 2.1: (I) föreställer den nuvarande animation som finns i Lydian. (II) föreställer vårt förslag till design av samma algoritm.

3 Metod

I concluded that I might take as a general rule the principle that all things which we very clearly and obviously conceive are true: only observing, however, that there is some difficulty in rightly determining the objects which we distinctly conceive.
Rene Descartes, Discours de la Methode, 1637

3.1 Litteraturstudier

Då vi i denna uppsats skall undersöka hur visualiseringar skall designas för att stödja inlärning måste vi få en uppfattning om hur inlärningsprocessen går till samt hur multimedia kan stödja denna process. Vi har valt att göra en litteraturstudie för att inhämta denna kunskap.

För att hitta material om den psykologiska och pedagogiska forskningen har vi utgått från Richard Mayers forskning, då han är en väletablerad forskare inom pedagogisk psykologi. Han är professor i psykologi vid University of California, har publicerat över 200 artiklar, främst inom utbildningspsykologi, skrivit 12 böcker och har varit ordförande för avdelningen för utbildningspsykologi i American Psychological Association. Vi har utifrån de referenser som finns i hans artiklar fördjupat oss i de teorier som ligger till grund för hans forskning samt forskning som är relaterad.

Då det gäller artiklar och annat material om forskning om algoritmvisualisering använde vi oss av artiklar som vi hittat på nätet genom att använda sökmotorer som Google[1] och AllTheWeb[2]. Vi använde oss också av ACM's[3] databas över artiklar som publicerats av dem i tidskrifter och konferensprogram. Vi fann att personer på Georgia Tech arbetat mycket inom det här området, främst John Stasko, och deras artiklar var också flitigt publicerade på andra ställen. Därför koncentrerade vi oss på denna forskning om algoritmanimationer.

De flesta artiklar är funna i tidsskrifter eller via internet, böcker har vi lånat på bibliotek eller av vänner och kollegor.

Då multimedia är något som utvecklas i snabb takt har vi koncentrerat oss på den forskning som har pågått på 1990-talet inom detta ämne. Psykologi har däremot inte utvecklats i samma takt så vi har där inte varit lika fokuserade på utgivningsår.

Artikelmaterialet om *Deterministic Distributed List Coloring* har vi fått av Phillipas Tsigas på institutionen för Datalogi, Chalmers Tekniska Högskola.

Utifrån materialet som vi samlat in genom litteraturstudien har vi sedan ställt upp ett antal krav som måste uppfyllas av en visualisering för att den skall kunna stödja inlärning.

3.2 Distribuerade system och algoritmer

I den diskussion vi för längre fram i uppsatsen om hur vi anser att visualiseringar skall designas för att fungera som undervisningshjälpmedel använder vi en distribuerad algoritm som utgångspunkt för designen. Av den anledningen följer här några definitioner och begrepp som vi hoppas skall underlätta förståelsen den diskussionen. Materialet är hämtat från Gerard Tel *Introduction to Distributed Algorithms* [42].

[1] http://google.com
[2] http://alltheweb.com
[3] Association of Computing Machinery, http://www.acm.org

Distribuerade system

Ett distribuerat system definieras av Tel som *en förbunden samling av självständiga datorer, processer, eller processorer*. Datorerna, processerna eller processorerna kallas för *noder* i det distribuerade systemet. För att anses vara självständiga måste noderna bestämma över sig själva, för att anses vara förbundna måste noderna kunna utbyta information.

I den distribuerade algoritmen vi kommer att visualisera kommunicerar noderna med hjälp av *message passing*, det vill säga att de använder den kommunikationskanal som förbinder noderna för att sända meddelanden. Denna kommunikationen mellan noderna ger oss ett *tidsbundet samband mellan händelser i systemet.* Nod *p* kan inte ta emot ett meddelande före nod *q* har skickat det.

Distribuerade algoritmer

En distribuerad algoritm för en samling processer är en samling av lokala algoritmer, en för varje process.

Tel använder ett antal begrepp för att beskriva egenskaper hos distribuerade algoritmer, de viktigaste för vår diskussion är dessa: *fairness, safety, liveness, causal order, complexity* och *failure locality.*

Fairness — Snällhet Med *fairness* avses hur snälla noder är mot andra noder i systemet. En algoritm med hög *fairness* åstadkommer att alla noder under algoritmens exekvering får tillgång till gemensamma resurser, därmed undviks så kallad *starvation. Starvation* inträffar när en process under den tid algoritmen exekverar aldrig får tillgång till en gemensam resurs.

Safety — Den är min! Detta begrepp syftar på behovet att synkronisera åtkomst av gemensamma resurser. En distribuerad algoritm arbetar ofta i en miljö där det är nödvändigt att synkronisera användningen av en gemensam resurs, till exempel datastrukturer. *Safety*-egenskapen skall entydigt visa att endast en nod använder en viss gemensam resurs vid en tidpunkt t_n.

Liveness — Jag ska bara. . . , sa Alfons *Liveness* betyder att en process inte terminerar före den uppnått det slutmål algoritmen definierat. Eftersom noden samtidigt är tvungen att ta hänsyn till andra noder på grund av *fairness*-egenskapen är det inte nödvändigtvis så att varje händelse noden upplever innebär att den är ett steg närmare målet. Det kan till och med innebära att en händelse för noden innebär att den förs ett eller flera steg längre ifrån målet. *Liveness* är i någon mening ett mått på att algoritmen är korrekt.

Causal Order — Hur mycket är *din* klocka? För vissa distribuerade algoritmer kan två eller flera meddelanden byta plats utan att påverka senare tillstånd i systemet. Detta innebär att tid som global storhet är otillräcklig för distribuerad exekvering och istället används begreppet *causal order*. *Causal order* definieras av Tel som:

Låt *E* vara en exekvering av en distribuerad alogritm. Relationen $\prec$, *causal order*, för händelserna i *E* är den minsta relation som tillfredställer:

1. Om *e* och *f* är olika händelser för en process och *e* inträffar före *f*, så $e \prec f$.
2. Om *s* är en sändhändelse och *r* är motsvarande ta-emot-händelse, så $s \prec r$.
3. $\prec$ är transitiv.

Causal order innebär att det under vissa omständigheter går att veta att en händelse inträffar före en annan sett utifrån en specifik nod. Vissa av händelserna en nod utsätts för under en exekvering har beroendeförhållanden, det råder förutom en kausal ordning dessutom ett kausalt beroende mellan händelserna. Händelser som är kausalt beroende av varandra är ofta intressanta ur den synvinkeln att det är förknippade med någon central aspekt av algoritmens arbete.

Complexity — Rörighet Komplexitet hos distribuerde algoritmer mäts i olika avseenden: *meddelande-*, *bit-*, *tids-* samt *minneskomplexitet*. Vi är intresserade av två av dessa, meddelande- och tidskomplexitet. Givet ett distribuerat system, en distribuerad algoritm och ett sluttillstånd är meddelandekomplexiteten det antal meddelanden som måste skickas mellan noderna för att dessa skall kunna tillse att slutmålet uppnås. Tidskomplexiteten är den tid[4] det tar att utföra en viss uppgift.

Failure locality — Dominoeffekt. Detta begrepp syftar på hur feltolerant algoritmen är. I en algoritm med låg *failure locality* påverkas få noder omkring en nod som slutar fungera.

3.3 Prototyp

Vi skall i denna uppsats designa en prototyp som senare kan komma att ingå i Lydian. Prototypen skall visualisera en distribuerad algoritm och förhoppningsvis göra det på ett sådant sätt att den faktiskt underlättar för studenterna att förstå hur algoritmen fungerar.

Den målgrupp prototypen riktar sig mot är samma målgrupp som Lydian riktar sig mot, studenter som deltar i en kurs om distribuerade algoritmer och system under samma förutsättningar som den ges på Chalmers Tekniska Högskola i Göteborg.

När vi har designat prototypen har vi utgått från de pedagogiska och psykologiska teorierna samt de krav som vi utifrån dessa ställt upp. Vi har även studerat den algoritm som skall visualiseras för att urskilja vilka delar och koncept som är relevanta samt ifall det är möjligt att dela upp den i flera animationer. Utifrån teorin har vi sedan hittat lämpliga sätt att presentera dessa delar och koncept på ett sådant sätt att de uppfyller de uppställda kraven.

[4] Med tid avses här en kedja av händelser med kausal beroenden.

4 Teori

Knowledge is only perception
Sokrates (470-399 fKr)

Vi skall i det här kapitlet gå igenom den teori som vi senare skall använda oss av. Vi börjar med en genomgång av utvalda delar av perceptions- och kognitionspsykologin och fortsätter sedan med den pedagogiska forskning som har gjorts om användning av animationer. Efter det skall vi presentera viss annan forskning som vi anser vara relevant för denna studie; forskning om algoritmanimationer och om färg som informationsförmedlare.

Då denna teori är presenterad formuleras ett antal krav som måste vara uppfyllda för att en visualisering skall stödja inlärning.

4.1 Kognition och perception

Ett antal försök att få studenter att bättre förstå algoritmer med hjälp av animeringar har uppvisat mestadels negativa resultat [3, 8, 21, 22]. Enligt Petre et al. [41] innebär förståelse för en algoritm att bygga associationer mellan händelser och entiteter i ett program och händelser och entiteter inom arbetsområdet. Hur bygger man då dessa associationer? För att försöka svara på denna fråga skall vi här ge en inblick i relevanta teorier inom perceptions- och kognitionspsykologin.

4.1.1 Perception

Enligt Encyclopedia Brittanica[12] är perception processen att ta emot och bearbeta information från omgivningen. Det är alltså genom de perceptuella systemen vi tar emot information och kan börja bearbeta den. Inom perceptionsteorierna finns också en stor mängd ideer som är direkt tillämpningsbara i animeringar [41]. Vi skall försöka ge en inblick i dessa perceptionsteorier.

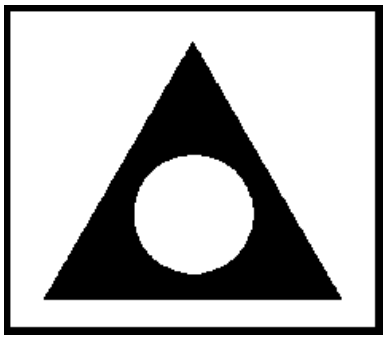
Gestaltteorin Gestaltteorin utvecklades i början av 1900-talet med rötter inom psykologi, logik och epistemologi[1] och försökte besvara frågan “Varför ser saker ut som de gör”[2]. Det fundamentala i gestaltteorin är följande: “Det finns helheter, vilkas beteende inte är bestämt av deras individuella element, men där delprocesserna själva är bestämda av den inneboende naturen av helheten. Det är Gestalt-teorins hopp att bestämma naturen hos sådana helheter.”[3]

När det gäller den visuella perceptionen ser gestaltteorin den som en dynamisk, organiserad process och ett fenomen som visar på det är hur vi skiljer på en figur och dess omgivning. En figur är oftast komplett, sammanhängande och framför sin omgivning som uppfattas mindre skarpt. Detta är en dynamisk process då vi i vissa sammanhang kan växla mellan att betrakta en viss del av bilden som bakgrund och som figuren(fig 4.1). Gestaltteoretikerna upptäckte även ett antal principer för hur grupper av synintryck organiserar sig [15]. De

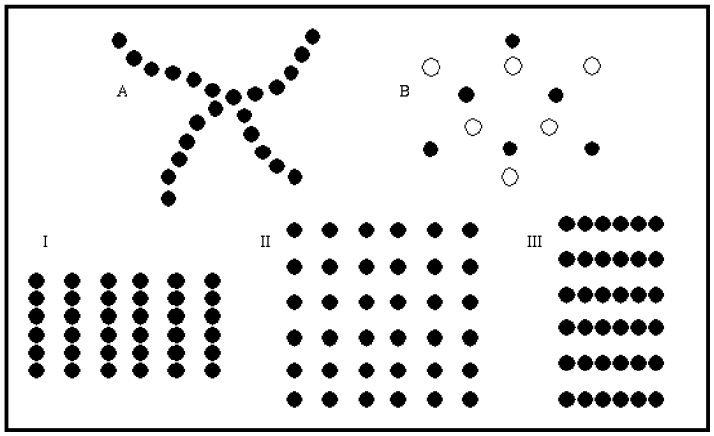
[1] Läran om naturen, ursprunget och gränserna för mänskligt vetande

[2] “Why do things look as they do” - Kurt Koffka[15]

[3] "Man könnte das Grundproblem der Gestalttheorie etwa so zu formulieren suchen: Es gibt Zusammenhänge, bei denen nicht, was im Ganzen geschieht, sich daraus herleitet, wie die einzelnen Stücke sind und sich zusammensetzen, sondern umgekehrt, wo - im prägnanten Fall - sich das, was an einem Teil dieses Ganzen geschieht, bestimmt von inneren Strukturgesetzen dieses seines Ganzen."[44]

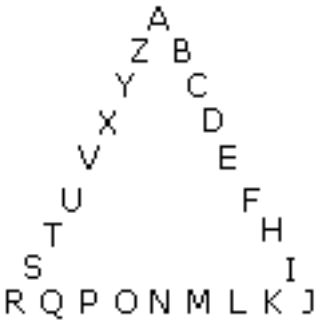


Figur 4.1: Mångtydigt figur-bakgrundförhållande. Ligger den vita cirkeln ovanför den svarta triangeln eller är det ett hål genom vilket bakgrunden kan ses?



Figur 4.2: (A) Gruppering efter kontinuitet (B) Gruppering efter likhet (I) Gruppering efter närhet gör att vi uppfattar gruppering i kolumner (II) Gruppering efter symmetri. Avstånden är lika och ingen dominant gruppering uppstår (III) Gruppering efter symmetri. Närheten gör att vi uppfattar gruppering i rader

klassificerade fyra typer av grupperingar : gruppering efter närhet, symmetri, kontinuitet och likhet (fig 4.2).



Figur 4.3: Ett exempel på gestalteffekten. Bilden är uppbyggd av bokstäver ur vilka den triangel vi uppfattar inte kan härledas.

Gestalteffekten Med gestalteffekten menas att en helhet uppfattas som mer än summan av sina delar [15, 25]. Denna effekt uppträder ofta inom visuell perception [15] , men även inom andra områden (se fig 4.3). Gestalteffekter uppträder inom all perception och även inom flera andra delar av psykologin och till och med fysik [25]. Det är en viktig effekt att uppnå eftersom det är kring helheter det går att resonera och dra slutsatser och ifall helheten inte har uppfattats blir slutsatserna lätt ofullständiga och resonemangen felaktiga.

Parallell och sekventiell bearbetning Paivio [39] karakteriserar den visuella perceptionen med dess spatiella egenskaper som utgångspunkt. Receptorerna och de högre neurala delarna av det visuella systemet är spatiellt organiserade och därför kapabla att ta emot, sända och bearbeta information simultant. Då de funktionella delarna är spatiellt och organisatoriskt parallella kan det visuella systemet betraktas som ett parallellt arbetande system. Dock noterar Paivio att även om själva systemet är parallellt så kan processen att bearbeta visuella intryck mycket väl vara sekventiell. Uppfattning och organisation av textur och färg sker till exempel parallellt men uppfattning och organisation av yta, form och inneslutning sker sekventiellt [26]. Paivio påpekar också att det auditiva perceptionssystemet till skillnad mot det visuella är sekventiellt.

4.1.2 Minnet

Förståelse för en algoritm innebär alltså att bygga associationer mellan objekt i programmet och objekt i arbetsområdet [41]. Vad är då dessa associationer, och hur lagras de i vårat minne? Vi får än en gång vända oss till psykologin för att försöka få svar på dessa frågor.

Minnestyper I litteraturen delas minnet upp i olika kategorier med avseende på vilken typ av innehåll minnet lagrar [27, 26, 1, 39, 40]. Lundh [27] delar in minnet i följande kategorier:

- Semantiskt minne — Det semantiska minnet är all vår organiserade kunskap om ord och andra verbala symboler, dvs i princip alla kunskaper vi har som går att formulera i ord.

- Perceptuellt minne — Det perceptuella minnet är våra kunskaper om objekts utseende, spatiala förhållanden i omgivningen, hur kaffe luktar osv. Dessa minnesuttryck är oftast svåra att uttrycka i ord.

- Procedurellt minne — Hit hör alla färdigheter som en person har lärt in, till exempel hur man cyklar, installerar Linux eller uttyder information ur en graf [26].

Mentala koder och dual-code teorin All information som assimileras av minnet måste lagras i någon form av inre representationer, så kallade kognitiva strukturer [27]. Paivio [39, 40] talar om att stimulus från omvärlden transformeras och omvandlas inom oss och när den yttre stimulansen försvinner finns det en kodad representation av stimulin kvar, en mental kod. En mental kod är alltså en inre representation av yttre data, exakt hur denna kod rent fysiskt finns lagrad är tämligen irrelevant för denna uppsats.

Paivio gör skillnad mellan koder som representerar verbal och bildlig information. Med verbal information menas i detta sammanhang information som implicit eller explicit involverar de audiomotoriska systemen, till exempel ett ord i skrift vilket utläses tyst. Bildlig information är information som inte involverar sådana system. Dessa olika typer av information representeras också i minnet av olika mentala koder, verbala och bildliga koder.

Experiment som Paivio har genomfört visar att det finns skillnader på hur dessa koder representerar information, till exempel är verbala koder bättre på att lagra procedurell information medan visuella koder är bättre lämpade för att koda spatiell information. Dessutom är verbala koder organiserade i termer av associationer och hierarkiska strukturer medan bildliga koder är organiserade i helhet-delar förhållanden.

Processen som kodar om information till dess inre representation är uppdelad i tre olika delprocesser på olika nivåer som är tätt sammanknutna med informationens mening eller betydelse:

1. **Den representationella processnivån** — Denna process innebär att informationen kodas om till en symbolisk representation som lagras i långtidsminnet som en konkret minnesbild. Ifall denna representation är en verbal eller bildlig kod är beroende på informationens innehåll. Detta är alltså den initiala lagringen när vi först blir medvetna av ett fenomen, vi kan aktivera minnet och komma ihåg att fenomenet existerar.

2. **Den referens-associativa processnivån** — På denna nivå sker en associativ reaktion som kopplar samman bildliga och verbala representationer av ett visst fenomen. Vi kan alltså koppla ihop de bildliga och verbala koderna med varandra, till exempel kan vi se en cirkel framför oss när vi hör detta ord och tvärtom.

3. **Associativa kedjor och strukturer.** På den här nivån skapas sekvenser eller mönster av associativa reaktioner mellan fenomen och koncept, detta gäller även skilda fenomen. Vi kan nu associera olika bilder och ord med varandra, vi kan till exempel koppla ihop cirklar med sfärer, kuber och andra geometriska konstruktioner.

Att det har skapats en association mellan de bildliga och verbala representationerna av samma fenomen gör att det är lättare för att erinra sig om det. Paivio har gjort experiment som visar att det är lättare att komma ihåg konkreta substantiv än att komma ihåg abstrakta sådana. Detta kan förklaras genom att konkreta substantiv associeras till bildliga representationer av objektet. Teorin om att det finns skilda bildliga och verbala koder som refererar till samma koncept med kopplingar mellan sig och att minnesåtkomsten gynnas av detta kallas dual-code teorin.

Korttidsminne Korttidsminnet, eller arbetsminnet, handlar om det som vi minns av den tidigare delen av det psykologiska nuet snarare än av det förflutna [20]. Dess uppgift är att behålla information som vi utför mentala operationer på, det spelar alltså en stor roll i samband med målinriktat tänkade och problemlösning [27]. Korttidsminnet har en mycket begränsad förmåga till informationslagring. G A Miller [36] visade att gränsen till hur mycket information som kan lagras tycks ligga runt 7 ± 2 meningsfulla enheter av information. Med meningsfulla enheter menas i detta sammanhang en enhet som har betydelse för den som uppfattar informationen. Detta innebär också att då nya enheter lagras i korttidsminnet, till exempel därför att man får nya perceptuella intryck, kommer den information som tidigare lagrades i korttidsminnet att överlagras och försvinna [1].

Långtidsminnet Långtidsminnet, det vi i dagligt tal kallar minne, är kännedom om en händelse eller fakta tillsammans med vetskapen om att vi har upplevt eller tänkt det tidigare [20]. Det fungerar alltså som en förvaringsplats för mentala representationer av olika typer av innehåll [1]. Det finns olika sätt att se distinktionen mellan lång och korttidsminne, antingen som två olika minneslager eller som att korttidsminnet är den för tillfället aktiva delen av långtidsminnet [27]. Vilket av synsätten som väljs är för oss irrelevant, vad som har betydelse är hur information kan fås att lagras i långtidsminnet. Det finns uppskattningar om att information behöver lagras i korttidsminnet i cirka fem till tjugo sekunder för att lagras i långtidsminnet [1]. Genom repetition, tyst eller uttalad, av informationen kan man informationens lagring i korttidsminnet förlängas och därmed öka minnesbehållningen [27, 11]. Att aktivt relatera information till redan känd kunskap ökar också minnesbehållningen [1]. Det har även visat sig att det går uppnå en högre grad av minnesbehållning genom att presentera information i båda verbal och bildlig form i kombination med att visa på representiella och referentiella samband mellan de olika representationerna [39, 28, 29, 30, 35].

4.2 Pedagogisk forskning

Richard Mayer, professor i psykologi vid University of California i USA, har länge bedrivit forskning om hur multimedia kan hjälpa studenter att ta till sig kunskap och använda kunskapen i nya sammanhang.

Mayers definition av multimedia har mer med olika medium att göra och mindre med datorer. Mayer pratar om multimedia som en sammansättning av två primitiva medium, till exempel text och bild.

Richard Mayers arbete utgår ifrån Paivios dual-code teori. Mayers forskning har syftat till att ge empiriska bevis för dual-code teorins riktighet och hur man bäst utnyttjar de faktum dual-code teorin postulerar. Vidare bygger Mayer sitt arbete på den kognitiva konstruktivismen, och dess påverkan på epistemologin.

Samtidighet

Mayer har i flera experiment visat på positiva effekter av att presentera information i två former, en för det verbala systemet och en för det icke-verbala. Försöken påvisar bättre förmåga att minnas det lärda och dessutom att använda den kunskapen i nya sammanhang [29]. För att uppnå den önskade effekten måste presentationen synkroniseras eller vara nästan synkroniserad [35]. Information presenterad i två former underlättar för människor att minnas informationen, vilket följer naturligt av dual-code teorin eftersom det finns två mentala representationer av samma information [28].

Den positiva effekten av att informationen presenteras i två former som ligger rumsligt eller tidsmässigt intill varandra har Mayer visat vara statistiskt säkerställd [28, 29, 35]. Mayer använder i alla sina försök tre typer av tester som försökspersonerna får svara på efter att de genomgått undervisningen:

1. Förmåga att minnas det lärda (retention test)
2. Förmåga att lösa problem baserat på det lärda (transfer test)
3. Förmåga att namnge element i en avbildning av det lärda (matching test)

Kognitiv belastning

I nära anknytning till betydelsen av att information presenteras samtidigt ligger betydelsen av att inte överlasta den kognitiva apparaten. George Miller gjorde i “The Magical Number Seven, Plus or Minus Two” en sammanställning av forskningsresultat som samstämmigt visade att kognitionsapparatens kapacitet är starkt begränsad [36]. Vidare visades att om försökspersonen kunde binda ihop flera mindre informationsbitar till större informationsenheter, kvarstod den gräns för hur många informationsenheter personen kunde ha medveten uppfattning om. Detta visar på vikten av att inte överlasta korttidsminnet. Mayer har dessutom visat i försök att möjligheten att skapa referentiella och representationella förbindelser enligt Paivios dual-code teori ökar om information presenteras på ett sådant sätt att både den information som skall till

det verbala och det icke-verbala systemen finns i arbetsminnet samtidigt [35]. Mayer har gjort försök där det fanns en grupp som fick information presenterad i två olika kognitiva former samtidigt; en grupp som fick samma information som grupp 1, men där det var en liten förskjutning mellan presentationen i de olika formerna; och en tredje som fick presentationen av information först i den ena kognitiva formen och sedan i den andra kognitiva formen. Resultaten för de olika grupperna visade att de två första grupperna presterar lika bra på retention, transfer och matchnings testerna, medans den tredje gruppen presterade sämre på alla tre testerna [28, 29, 30, 35].

Multimedia

Med multimedia avses att information presenteras i två eller flera olika presentationsformer. Mayer har preciserat begreppet genom att se på multimedia ur tre synvinklar [32]:

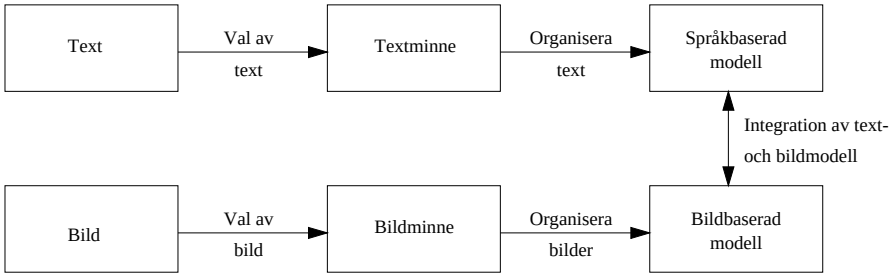
- Levererande medium — papper, datorskärm
- Presentationsform — ord, bild
- Sensorisk form — detta syftar på vilka sinnen den som uppfattar informationen använder för att ta in den presenterade information

Mayer argumenterar för att skillnaden i att använda olika medium för att leverera multimediapresentationen är liten [32]. Möjligheten för en lärande person att skapa nödvändiga förbindelser mellan verbala och icke-verbala representationer beror alltså inte på vilket medium som används. I flera försök har Mayer, som redan nämnts, visat på de positiva effekterna av att få information presenterad i flera olika former [28, 29, 30, 35]. Det verkar dock som om personer med liten kunskap om det presenterade har större nytta av en multimediapresentation än personer med stor kunskap [30, 32]. Vidare är multimedia ett bättre hjälpmedel för personer med hög spatial förmåga än för personer med låg spatial förmåga [30].

Mayer har i några inledande försök visat att den sensoriska formen kan få stor betydelse. Om den verbala informationen är *skriven* text konkurrerar texten och bilderna om den visuella uppmärksamheten, vilket ökar risken att information går förlorad [32].

Konstruktivistiskt lärande

Den kognitiva konstruktivismen ser på kunskap utifrån den lärande. Kunskap kan inte överföras mellan två personer, utan den som lär sig måste själv *skapa* kunskapen [19, 35]. Att lära är en kunskapsskapande process [19]. Processen att skapa kunskap går till så att den lärande assimilerar sina upplevelser av världen med sin existerande förståelse av den [19]. Först organiserar den lärande den nya informationen i kausala samband, vad Mayer kallar interna förbindelser, därefter integreras den nya informationen med tidigare kunskap, det som Mayer kallar externa förbindelser [35]. För att en kunskapsskapande



Figur 4.4: Processen att välja informationsbitar, organisera dem och integrera verbal och visuell modell.

process skall äga rum argumenterar Mayer för att den lärande *måste* genomgå dessa steg [32]:

- aktivt välja relevant information
- organisera och integrera den till en samstämmig mental representation
- integrera dessa representationer med redan existerande

Inom ramen för dual-code teorin kan vi se hur de två första stegen går till (fig 4.4). Vi tänker oss en situationen där information presenteras med text och bild. Text och bild väljs av de verbala och visuella systemen och lagras i text- respektive bildminne. Intrycken organiseras i en verbal och en visuell modell. Om information som beskriver samma fenomen finns som verbal och visuell modell samtidigt kan man skapa en integrerad mental modell av fenomenet.

4.2.1 Föreslagna designprinciper (Mayer)

Mayer har tagit fram fem desingprinciper som syftar till att stödja möjligheten till konstruktivistiskt lärande. Mayer anser att de har positiv effekt med avseende på inlärning [34].

1. Multimedia — Information skall presenteras i former som ger både det verbala och det visuella kognitiva systemen möjlighet att bearbeta den [28, 29].
2. Kontinuitet — Information skall presenteras så att det är möjligt för den lärande att ha både den verbala och den visuella delen i korttidsminnet samtidigt [28, 29, 30, 37].
3. Relevans — Presentera enbart relevant information. Undvik information som är relaterad men inte nödvändig [16, 17, 31].
4. Presentationsform — Den verbala informationen ger lägre kognitiv last om den presenteras som tal än som text [33].
5. Individuella skillnader — Multimedia hjälper personer med liten kunskap om det presenterade materialet bättre än de som redan har god

förståelse för materialet. Samt att personer med låg spatial förmåga har mindre nytta av multimediapresentationer än personer med god spatial förmåga [30].

4.3 Animation av algoritmer

Då vi samlade in litteratur till detta arbete sökte vi litteratur om algoritmanimation. Forskningen om värdet av algoritmanimation som undervisningshjälpmedel är en föga uppmuntrande läsning. Generellt går det endast påvisa en svag positiv effekt av att använda animationer [2, 8, 22]. I undersökningar där studenterna fått göra egna animationer är resultaten mer positiva [4]. Petre *et al* påpekar att en del positiva resultat kan komma av den roligare undervisningsmodellen, och att lika positiva resultat kunnat åstadkommas med andra mer underhållande undervisningsmetoder än traditionell föreläsningsmodell [41].

Christopher Hundhausen har i sin doktorsavhandling [19] riktat kritik mot en del av dessa arbeten. Han angriper den teori[4] som han ser som grund till många av de försök som gjorts vad gäller animation som undervisningshjälpmedel för att lära ut algoritmer. Hundhausen argumenterar för att många försök med animationer av algoritmers arbete utgår ifrån en experts syn på algoritmen och att den bärande iden är att koda expertens mentala modell i animationen och att den sedan avkodas av studenten som då förmodas få samma representation som experten i sitt huvud.

Det finns en stark tro hos experter på att de har en korrekt mental bild, och att om de bara lyckas förmedla den, kommer andra att förstå. Experter tycker själva inte om andras visualiseringar [41], utan gör istället egna eller gör om dem som andra gjort.

4.4 Färg som informationsförmedlare

Det mänskliga ögat är mycket känsligt för färgvariationer. En tränad människa kan skilja mellan 1 000 000 olika färgnyanser i tester med parvisa jämförelser och ungefär 20 000 nyanser kan utskiljas av många otränade människor [43]. För att representera abstrakt information ger dock mer än 20-30 olika färger negativa resultat [43].

Inom management informationsystems forskning har man studerat användadet av färger i grafiska presentationer och man har kommit fram till att [18]:

- färger förbättrar prestationen i återerinringsuppgifter (recall tasks).
- färger förbättrar prestationen i sök och lokaliserings uppgifter (search-and-locate tasks).

[4] Hundhausen benämner denna teori “Epistemic Fidelity Theory” (EF). Det är framförallt tre saker han angriper. (i) EF ser på kunskap som en representation i människors huvuden. (ii) Enligt EF kan kunskap om algoritmer förflyttas mellan en experts medvetande och en algoritmstudent via en animation. (iii) EF hävdar att grafiska representationer, t ex animationer, ligger nära en experts mentala modell av en algoritmer. [19]

- färger förbättrar prestationen i minnesbehållnings uppgifter (retention tasks).
- färger ökar förståelsen av instruktionsmatriel.
- färger förbättrar prestationen i besluts avvägnings uppgifter (decision judgement tasks).

Dock påpekas det också att färg är en subtil variabel som kan ge stora positiva effekter om den används rätt, men okritiskt användande av färg ger inte automatiskt resultat [18]. Färger kan bland annat användas för att markera aktivitet, betona mönster och visa olika tillstånd [7].

4.5 Designimplikationer

Ur den teori som vi presenterat fastställer vi fem designkriterier som vi använder för att validera den design vi själva föreslår i kapitel 5.

- Enkelhet — Då korttidsminnets kapacitet är begränsad är det viktigt att information på ett enkelt sätt kan utläsas ur visualiseringen. Om det krävs att användaren skall använda en teckenförklaring för att förstå innebörden krävs en ständig förflyttning av uppmärksamheten vilket leder till att korttidsminnets innehåll skrivs över. För att undvika detta bör information presenteras med välkända symboler eller självklar symbolik.
- Fullständighet — Förståelsen av en algoritm kräver att det finns en förståelse för varje i algoritmen ingående beståndsdel. Detta kan medföra att visualiseringen måste delas upp för att det skall vara möjligt att uppfatta algoritmens beståndsdelar.
- Kausalitet — För att kunna organisera och integrera kunskap om algoritmer måste det vara möjligt att följa kausala orsaksförhållanden i algoritmen. Detta underlättas genom att tydliggöra de samband som är centrala för algoritmen. Samband kan illustreras med hjälp av till exempel gestaltteorins principer om gruppering eller genom färgkodning.
- Korrekthet — Förståelse av distribuerade algoritmer kräver att representationer av specifika fenomen i algoritmen avbildas på ett korrekt sätt dels avseende kausalitet i algoritmen och dels avseende de representationer människor normalt kan antas ha om fenomenet.
- Överblickbarhet — Förståelsen av en distribuerad algoritm berör dels enskilda händelser och dels samverkan mellan olika delar i algoritmen eller den miljö algoritmen verkar i. Det betyder att det skall vara möjligt att uppfatta algoritmen och dess miljö som en helhet. För att det skall vara möjligt att utläsa en helhet i en visualisering måste designen ta hänsyn till korttidsminnets begränsningar. Med anledning av dessa begränsningar skall relaterad men inte nödvändig information undvikas.

5 Prototyputveckling

Welcome to the real world
Morpheus, The Matrix, 1999

I detta kapitel skall vi applicera den teori vi har presenterat i föregående kapitel på en prototyp som skall visualisera en distribuerad algoritm. För att under designen kunna föra en diskussion om våra designbeslut tänkte vi först presentera algoritmen i fråga.

5.1 Deterministic Distributed List Coloring

Deterministic Distributed List Coloring (DET-DLC) är en algoritm som är designad för att användas i mobiltelefonbasstationer för allokering av frekvenser. Den är utvecklad och föreslagen av Naveen Garg, Marina Papatriantafilou och Philippas Tsigas [14]. Det ligger utanför denna uppsats målsättning att i detalj förklara DET-DLC-algoritmen, utan vi kommer under följande diskussion enbart förklara de koncept som är centrala för förståelsen av algoritmen gentemot vårt problemområde, vilket dessutom kommer att ske löpande efterhand som begreppen behövs. Problematiken som algoritmen är tänkt att arbeta i är följande: För att mobil kommunikation med mobiltelefoner skall vara möjlig behöver varje telefon under ett samtal en frekvens att kommunicera med basstationen på. Två intilliggande basstationer får inte heller använda samma frekvens för kommunikation, om det sker kommer mobiltelefonsamtal att störas av varandra. Det ställer krav på synkronisering av basstationer.

DET-DLC är alltså en *distribuerad* lösning på problemet att allokera frekvenser till mobiltelefonbasstationer. Först och främst gör algoritmen det den ska, allokera frekvenser. För att det skall kunna ske på ett tillförlitligt sätt tillgodoser algoritmen dessutom kravet på synkronisering mellan basstationer, när de skall välja frekvens för att möjliggöra den mobila kommunikationen. DET-DLC har dessutom avsevärt mycket bättre *failure locality* jämfört med tidigare lösningar [14].

DET-DLC använder tre meddelanden, förutom de som härör från synkroniseringsmetoden, dessa är:

1. *confirm* — Basstationen har valt frekvenser och sänder meddelande till sina grannar om vilka den valt.

2. *acknowledge* — Basstationens grannar svarar med detta meddelande när de fått *confirm*.

3. *release* — En basstation sänder *release* när de slutar använda en frekvens.

5.1.1 Beskrivning av Choy-Singh algoritmen

Den synkroniseringsmetod som används i DET-DLC är en synkroniseringsalgoritm som föreslagits av Manhoi Choy och Ambuj Singh.

Choy-Singhs synkroniseringsalgoritm kan med rätta kategoriseras som icketrivial, av den anledning följer en icketrivial beskrivning. Denna beskrivning bygger på Manhoi Choy och Ambuj Singhs *Efficient Fault-Tolerant Algorithms for Distributed Resourse Allocation* [9].

Choy-Singh är en resursallokeringsalgoritm och den garanterar att bland processer som konkurrerar om samma resurs är det bara en som tillåts använda resursen vid ett given tidpunkt t_n. Vidare garanterar algoritmen *starvation freedom* och tidsfördröjningen från det att en process först begär den gemensamma resursen tills den får använda den gemensamma resursen är kvadratiskt beroende av antalet processer den konkurrerar med om resursen.

Ett nätverk av processer som skall synkroniseras med Choy-Singh algoritmen initialiseras först genom att varje process väljer en prioritet som inte får vara samma som någon av sina grannprocessers. Dessa prioriteter används för att lösa konflikter mellan grannprocesser när dessa skall använda den gemensamma resursen.

Choy-Singh använder en teknik som kallas *dörrar* för att åstadkomma de tidigare beskrivna egenskaperna. Det finns i deras algoritm två dörrar, en som processerna kan gå igenom asynkront och en som de kan gå igenom synkront. Processerna meddelar varandra när de passerar dörrarna, genom att skicka en signal m_1 när de går igenom den asynkrona dörren, signal m_2 när de lämnar den asynkrona och går in i den synkrona dörren och slutligen signal m_3 när de lämnar den synkrona.

Varje process har en lista med det senaste meddelandet som den fått från sina grannprocesser och när en process vill gå igenom den asynkrona dörren väntar den tills den fått en annan signal än m_1 från sina grannar. För den asynkrona dörren räcker det att titta på det senaste meddelandet från varje process en gång, om process *q* senast sänt m_3 kontrolleras *q's* meddelande inte igen. När det villkoret är uppfyllt kan processen gå igenom den asynkrona dörren och sänder själv m_1 till sina grannprocesser.

För den synkrona dörren måste alla grannprocesser synkront ha sänt ett annat meddelande än m_2. När detta villkor är uppfyllt kan processen gå igenom den synkrona dörren och själv sända m_2 till sina grannprocesser. Innan processen nu kan använda den gemensamma resursen måste den samla in tillåtelse från sina grannprocesser att göra det, m_2 är både ett meddelande som talar om att processen har gått in i den synkrona dörren och att processen vill ha tillåtelse att använda den gemensamma resursen. Schemat för att ge en grannprocess tillåtelse att använda den gemensamma resursen går till så här: Om en grannprocess *q* befinner sig någon annanstans än förbi den synkrona dörren ger den alltid sin tillåtelse för process *p* att använda den gemensamma resursen. Om process *q* däremot också befinner sig förbi den synkrona dörren ger den bara tillåtelse om process *p* har högre prioritet. Anledningen till att två processer som är grannar båda kan komma förbi den synkrona dörren är att detta är ett meddelandeskickande system och två processer kan uppfatta att villkoret för att gå igenom den synkrona dörren är uppfyllt och bägge gå igenom och sedan sända m_2 till varandra.

5.2 Prototypdesign

Det finns alltså flera olika nivåer man kan betrakta DET-DLC på. Vad en visualisering av DET-DLC *måste* klara av att visa är

- Synkronisering
- Frekvensallokering
- Algoritmens *failure locality*
- Grad av resursutnyttjande

Vi har dock valt att inte visa tidskomplexiteten då det är en egenskap som bestämms av Choy-Singh synkroniseringen och det är bevisat att en nods tidskomplexitet är kvadratiskt beroende av antalet grannar till en specifik nod. Det är alltså en bevisad egenskap och vi anser inte att den är relevant att visa för att förstå hur DET-DLC fungerar.

I enlighet med dual-code teorin och Mayers designprinciper måste denna animation presenteras parallellt med en berättarröst. Om animationen används i en föreläsningssituation är detta föreläsaren som beskriver det som sker i animationen, i en självstudiesituation en synkroniserad berättarröst. Vi har valt att dela upp visualiseringen i tre separata delar av tre anledningar:

- Kontinuitet — Då DET-DLC är en komplex algoritm med många relevanta faktorer som måste visualiseras är det omöjligt att presentera dessa i en enda animation på ett sådant sätt att den lärande kan ha både den verbala och visuella delen i korttidsminnet samtidigt.
- Relevans — Innan den lärande har förstått de grundläggande delarna av algoritmen saknar *failure locality* och resursutnyttjande relevans för personen i fråga. Så länge det inte finns en grundläggande förståelse för vad algoritmen gör är det ointressant hur bra den gör det.
- Naturligt snitt — Det finns ett “naturligt snitt” i algoritmen, vilket nämnts tidigare, nämligen mellan synkroniseringsförfarandet och frekvensallokeringen.

Vi får alltså tre stycken animationer: en som visar Choy-Singh algoritmen för synkronisering, en som utvecklar denna och introducerar hur frekvensallokeringen går till samt en som beskriver *failure locality* och resursutnyttjande.

Alla tre animationer presenterar endast information som är relevant för den förståelse som skall uppnås med animationen i fråga. I de två första animationerna skall den verbala informationen ha en nod som utgångspunkt och förklara dess beteende och vilka händelser som orsakar det. Vi behöver alltså inte ha särskilt många noder i dessa visualiseringar vilket får till följd att varje nod kan ta upp ganska mycket plats. För att ge en bild av algoritmens *failure locality* och resursutnyttjande behövs dock ett stort antal noder och den verbala

informationen måste här ha hela nätverket i fokus. Detta då *failure locality* och resursutnyttjande är egenskaper hos nätverket som inte enkelt kan härledas ur de individuella noderna.

Eftersom korttidsminnet endast kan innehålla ett fåtal enheter av information och dessa enheter "skrivs över" måste det gå att uttyda information utan att behöva betrakta ett stort antal objekt. Därför bör varje objekt innehålla all relevant information om objektet och helst på ett sådant sätt att det inte behövs hjälp av någon teckenförklaring för att uttyda informationen. Då det är relativt mycket information som skall presenteras samtidigt har vi valt att utnyttja mycket färgkodning av information då färger bearbetas snabbare än till exempel former.

5.2.1 Korrekthet för visualiseringar av distribuerade algoritmer

Med anledning av egenskaper hos distribuerade system (se sid 17) är det omöjligt att skapa en representation som är fullständigt konsistent med den verkliga exekveringen av algoritmen. Det som går att skapa är en representation som inte är felaktig avseende kasuala samband i algoritmen. För att tillförlitligt skapa en sådan representation måste animation vara konsistent avseende kausala samband i den verkliga exekveringen [38].

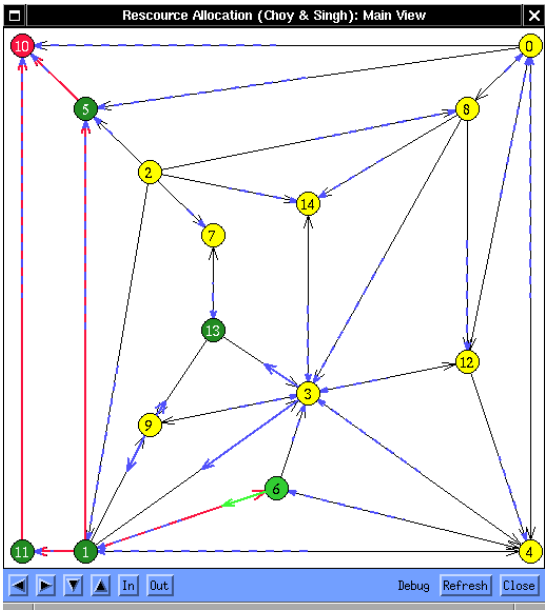
5.2.2 Choy-Singh

Det existerar idag en animation av Choy-Singh algoritmen i Lydian. Vi har valt att inte använda den existerande animationen utan istället att göra en ny. Vi tänkte börja med att beskriva den existerande animationen och framföra vår kritik mot den för att motivera detta. Därefter följer en beskrivning av hur vi designade en animation av algoritmen.

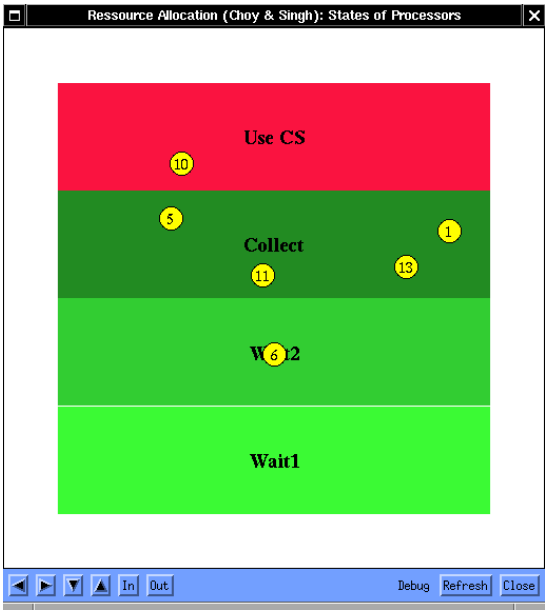
Beskrivning av den nuvarande animationen

Choy-Singh animationen i Lydian består av ett nätverksfönster (se fig 5.1) som visar nätverket som algoritmen opererar i och ett tillståndsfönster (se fig 5.2) som beskriver tillstånden för de olika noderna i nätverket. Utöver dessa finns de fönster gemensamma för alla animationer i Lydian (se sid 11). Nätverksfönstret visar samma sorts nätverk som alla andra animationer i Lydian. När en nod ändrar tillstånd ändras dess färg för att reflektera detta. Tillståndsfönstret är indelat i fyra olikfärgade fält och noder som representeras av cirklar rör sig mellan dessa fält då de byter tillstånd. Samma färger används i nätverks- och tillståndsfönstret för att visa samma tillstånd.

Meddelandena som utbyts mellan noderna symboliseras av en pil utgående från avsändarnoden. Denna pil förlängs tills den når halvvägs till mottagarnoden då den krymper tillbaka igen. Denna förlängning och förkortning av meddelandepilen fortgår tills medelandet har nått fram till mottagaren. Meddelandena har den färg som representerar det tillstånd som en nod har när meddelandet avsänds.



Figur 5.1: Nätverksfönstret i den nuvarande Choy-Singh animationen i Lydian



Figur 5.2: Tillståndsfönstret i den nuvarande Choy-Singh animationen i Lydian

Evaluering av den nuvarande animationen

Vi anser att valet av en pil som ändrar längd som representation av ett meddelande som sänds mellan två noder är svårtydbart. Förutom att ha en innebörd representerar meddelanden också två stycken momentana händelser i systemet, avsändandet och emottagandet av meddelandet. Exakt när dessa händelser sker blir otydligt när meddelandet representeras av ovan beskrivna pil vilket gör att den kausal ordning och eventuella beroenden blir svåra att avläsa. När två meddelanden skickas mellan samma två noder och det sist skickade skickas innan det första har kommit fram till mottagaren kommer dessa två att ligga ovanpå varandra vilket gör att endast ett av dem kommer vara synligt. Man missar helt det andra meddelandet.

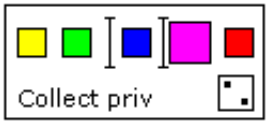
Enligt beskrivningen av Choy-Singh animationen i Lydian's hjälpfiler skall tillståndsfönstret hjälpa användaren att se hur lång tid processer befinner sig i vissa tillstånd samt vilka processer som hindrar sina grannar att fortsätta till nästa tillstånd. Enligt vår uppfattning misslyckas detta helt. Tillståndsfönstret innehåller ingen information som inte finns i nätverksfönstret. Dock presenteras informationen på ett sätt som är mindre lämpat för de förutsatta uppgifterna.

För att bestämma ifall en process blir hindrad av sina grannar från att fortsätta till nästa tillstånd krävs att man först lokaliserar noden i nätverksfönstret. Därefter skall man observera vilka noder som är grannar till denna nod och förflytta uppmärksamheten till tillståndsfönstret. På grund av bristen på struktur i detta fönster måste man inspektera ett antal noder innan man hittar någon av de sökta grannarna, ifall någon av dem överhuvudtaget finns med i tillståndsfönstret, och kan avgöra ifall det är denna som hindrar noden från att fortsätta. Med tanke på korttidsminnets begränsningar är det troligt att det är nödvändigt att under denna process gå tillbaka och friska upp minnet om vilka noder som var intressanta. Det är en betydligt mindre belastande process att direkt i nätverksfönstret avläsa de intilliggande nodernas tillstånd.

Vår animation av Choy-Singh

Vi skall nu presentera och motivera en design av Choy-Singh animationen som adresserar de brister som finns i den nuvarande animationen. Vi skall hela tiden följa de generella principer som vi diskuterade tidigare, och hela tiden fokusera på att ge användaren en grundläggande förståelse för hur Choy-Singh algoritmen fungerar.

Noderna I noderna (se fig 5.3) har vi valt att representera nodens möjliga tillstånd som fem olikfärgade kvadrater där den kvadrat som representerar det nuvarande tillståndet är större än de andra. Anledning till att vi vill visa alla möjliga tillstånden är för att betona den ordning i vilken tillstånden uppträder, och att vi vill skapa associativa kedjor mellan de olika tillstånden. För att presentera tillstånden även i en verbal form har vi också en textuell beskrivning av de olika tillstånden som visas under tillståndskvadraterna. Då vi förutsätter att användaren skall ha viss kännedom om algoritmen innan animationen



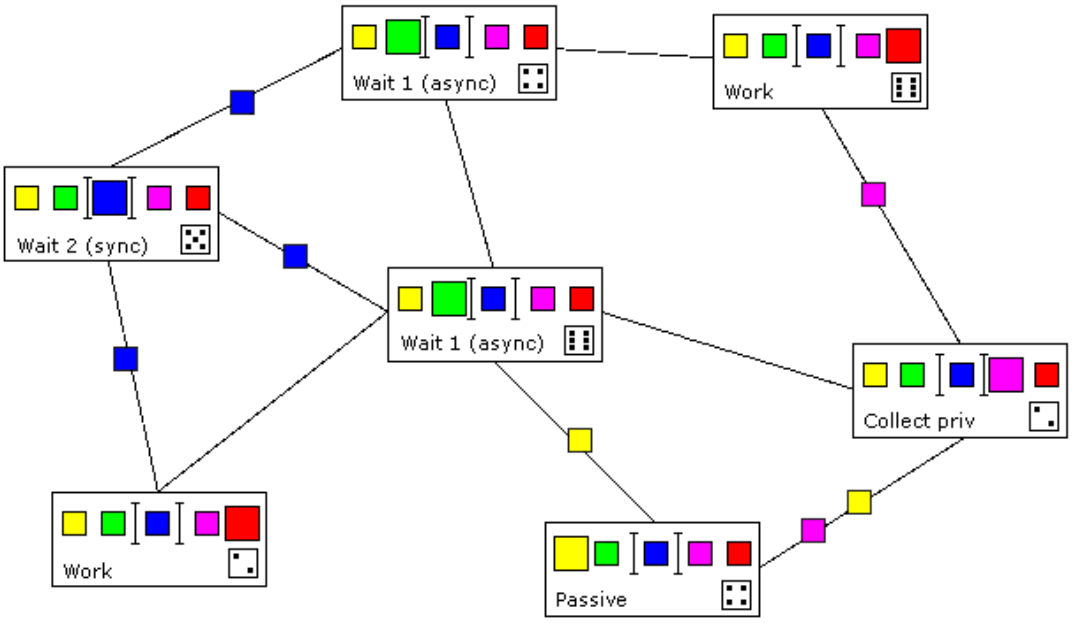
Figur 5.3: En nod till Choy-Singh visualiseringen.

används gör den textuella beskrivning även att färgkodning relateras till redan känd kunskap. Vi har även valt att visa de båda dörrarna som används som metafor i algoritmbeskrivningen av denna anledning.

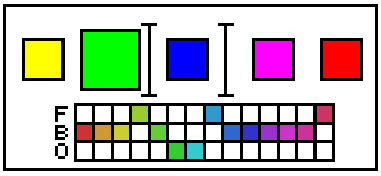
Det som finns kvar att visa i noden nu är nodens prioritet. Prioriteten har ett statiskt värde som passar att representeras i numerisk form. Att använda en siffra som visualisering av den skulle dock vara olyckligt då många andra animationer använder siffror för att numrera de olika noderna. Vi bestämde oss för att visa prioriteten som ett piktogram av en tärning då användaren kan förutsättas ha använt tärningar tidigare och således vet hur man uttyder information ur en sådan. Detta gör att avkodningsprocessen blir snabb.

Meddelanden För att tydligare visa när ett meddelande skickas iväg och tas emot har vi valt att visa meddelanden som kvadrater som med jämn hastighet rör sig från avsändarnoden till mottagaren. Denna representation stämmer också bättre överens med de flesta människors[1] kodning av begreppet meddelande, vilket gör det lättare att direkt inse att det är ett meddelande som skickas. Därmed blir algoritmens kausala ordning också lätt avläsbar ur animationen. Då meddelandena kommer att ta upp betydligt mindre plats i denna animation minskar också risken för att meddelanden överlagras och döljs av nya meddelanden. Färgen på meddelandena är identisk med de färger som representerar det tillstånd som avsändarnoden inträder i vid avsändandet (se fig 5.4). Skälet till att vi har valt samma färg och form på meddelanden som representationen av tillstånd är att tillstånd och meddelanden är tätt sammanknutna och vi vill underlätta för användaren organisera informationen på ett för algoritmen relevant sätt.

[1] Att de flesta människor föreställer sig ett meddelande som något som skickas mellan en avsändare och en mottagare snarare än som något som pendlar fram och tillbaka för att plötsligt försvinna är en helt egen subjektiv uppfattning. Vi tror dock att det är svårt att bestrida.



Figur 5.4: Ett helt nätverk av noder med meddelanden.



Figur 5.5: En nod i visualiseringen av DET-DLC på nodnivå.

5.2.3 DET-DLC

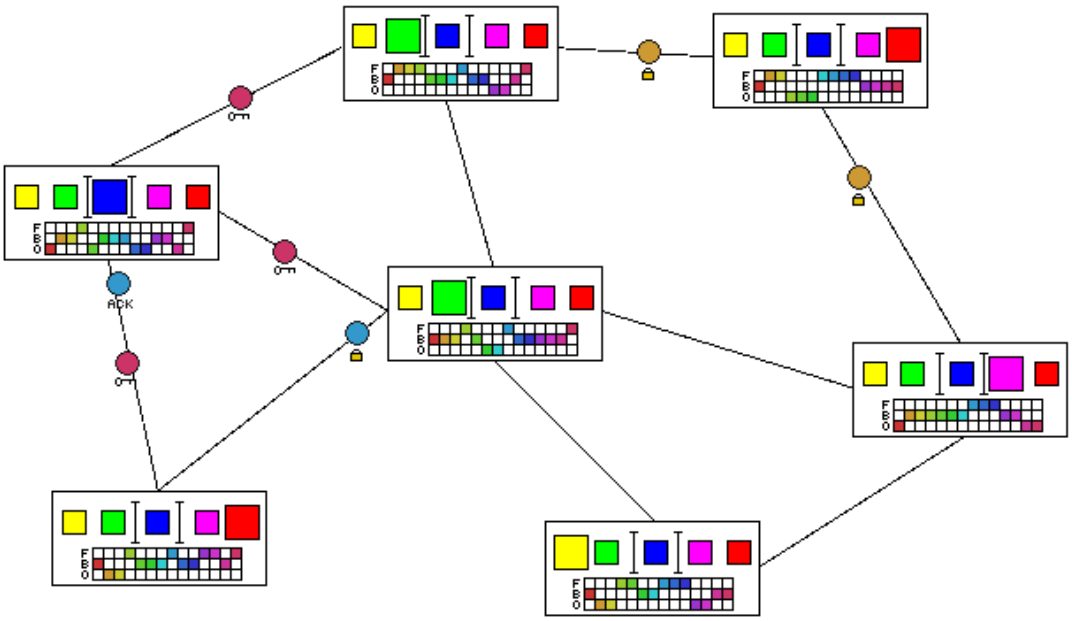
Visualiseringen av DET-DLC är, vilket nämnts tidigare, uppdelad i två animationer. Först skall vi presentera en animation på nodnivå, som skall förmedla hur allokeringen av frekvenser går till, och sedan beskrivs en animation på nätverksnivå för att presentera de övergripande egenskaperna hos algoritmen.

Animation på nodnivå

Då användaren i det här skedet redan har använt sig av Choy-Singh animationen har det lagrats information om hur man avläser information ur en sådan graf i det procedurella minnet. För att underlätta för användaren att uttyda information ur den nya animationen vill vi därför representera information på ett liknande sätt som i den förra animationen.

Noderna Vi vill ha samma representation av noderna i denna animation som i Choy-Singh animationen, men vi måste också ha en representation av vilka frekvenser som används av olika noder. Varje nod måste kunna visa vilka frekvenser den använder, vilka frekvenser som är lediga och vilka som är upptagna av andra noder eftersom dessa tre mängder av frekvenser är de som algoritmen behandlar. Då det i Choy-Singh animationen skall ha skapats referentiella associationer mellan de verbala och de bildliga koderna för tillstånden kan vi här ta bort den textuella beskrivningen av tillstånden för att undvika att noderna blir allt för stora (se fig 5.5).

När resursproblem diskuteras brukar resurserna ofta beskrivas som olika färger (se t ex [14]). Då detta är ett typiskt resursproblem där resurserna i fråga är frekvenserna har vi valt att representera de olika frekvenserna som färger. Genom detta val får vi också användaren att relatera animationen till andra algoritmer som använder sig av färger, något som kan förutsättas att denne har stött på i en algoritmkurs. Vi vill också att användaren skall kunna se att frekvenser omväxlande är fria, används av noden eller används av en annan intilliggande nod och har därför valt att i varje nod ha tre liggande listor med färger representerande av de olika mängderna liggandes bredvid varandra. Varje lista är märkt med en bokstav för att visa vilken av mängderna det är. På detta sätt kan användaren lätt se när en viss färg flyttas från en mängd till en annan. Eftersom vi vill introducera de koncept algoritmen bygger på och inte presentera en riktig situation skall det inte finnas mer än 15–25 färger att välja på i detta stadie.



Figur 5.6: Ett nätverk i DET-DLC animationen på nodnivå. Då confirmation meddelanden ’låser’ en färg betecknas de med ett hänglås, och release meddelanden med en nyckel.

Meddelanden I Choy-Singh animationen skall användaren ha fått en god förståelse av de olika meddelanden som ingår i synkroniseringen av noderna och därför skall vi här endast visa de meddelanden som är specifika för DET-DLC. Då vi skall visa meddelanden som har en annorlunda betydelse än de som förekom i den föregående animationen har vi valt att göra dessa meddelanden som cirklulära istället för kvadratiska (se fig 5.6). För att visa vilken typ av meddelande som skickas innehåller varje meddelande en symbol som kan ses som en metafor för meddelandets innebörd. Med detta vill vi dels att användaren direkt skall förstå meddelandets funktion och dels ge ett mindre abstrakt, känt, objekt eller koncept att associera funktionen med. I de fall där symbolen är ett konkret objekt liknar också användningsområdet meddelandets funktion, vilket vi hoppas skall ge upphov till associativa kedjebildningar. Meddelandena kommer också att ha samma färg som den frekvens meddelandet behandlar för att användaren lättare skall kunna associera en frekvens vandring mellan de olika frekvensmängderna med de meddelanden som sänds och tas emot av en nod.

Animation på nätverksnivå

Det tredje steget i visualiseringen av DET-DLC skall visa hur algoritmen fungerar på applikationsnivå. De två första visualiseringarna skall ge förståelse för synkroniseringsprocessen och frekvensallokeringsarbetet. Denna sista visualisering skall presentera algoritmen i ett större sammanhang. Visualiserin-

gen är mer komplex än de tidigare, men från dessa skall studenten nu ha förståelse för hur både synkroniseringen och frekvensvalet fungerar.

Vi vill att användaren skall få en uppfattning om hur nätverket beter sig under olika last, att det skall synas att de tyngst belastade noderna också får flest frekvenser. Det skall gå att se att algoritmen utför den funktion som den lovar.

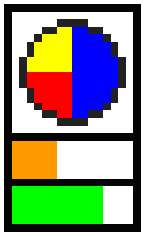
En ytterligare målsättning med denna animation är göra det möjligt att studera hur variationer av algoritmen påverkar utnyttjandegrad. I den kontexten kommer de två tidigare animationerna att utgöras av för den varierade algoritmen relevanta visualiseringar.

Failure locality visas genom att en eller flera noder sätts ur funktion och effekterna kan då iaktagas.

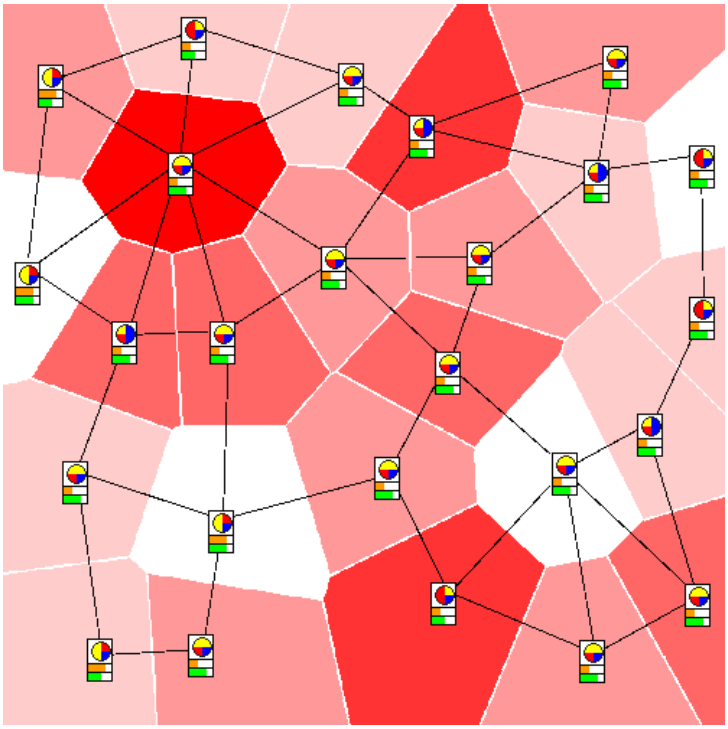
Nätverket Nätverket visar vilka noder som konkurrerar om frekvenser samt hur stor belastningen är på varje nod. Vi har valt att inte visa några meddelanden i denna animation då vi vill att användaren skall koncentrera sig på hur de olika noderna samverkar för att få en förståelse för helheten. Förbindelsen mellan noder visas som tidigare, med linjer som förbinder olika noder. Detta för att användaren skall känna igen att det är ett nätverk det handlar om och inte fristående noder. Belastning visas genom att bakgrunden varierar i färgton från vitt till röd. För att avgränsa det område i bakgrunden som hör till en viss nod ritar vi också upp ett *voronoi-diagram*[2] runt noderna. Valet av färg som varierar från vitt till rött är gjord då rött är en färg som ofta används som varning för att något är överbelastat, felaktigt eller varmt.

Noderna Denna visualisering kommer att innehålla avsevärt många fler noder än de tidigare. Målsättningen med visualiseringen är att ge förståelse för hur frekvensallokering sker i systemet efterhand som lasten varierar för olika noder (se fig 5.7). Noderna i denna visualisering skall visa hur många lediga frekvenser noden har tillgång till, samt hur många frekvenser noden använder. I animationen av nodernas arbete visas *vilken* frekvens som tillhör en viss nod, nu intresserar vi oss bara för hur *många* frekvenser noden använder och hur *många* den har lediga. Då vi endast är intresserade av antalet färger i varje mängd visas antalet frekvenser som staplar som växer och krymper. Varje stapel är märkt med samma piktogram som motsvarande mängd i animationen på nodnivå. Användaren känner då lätt igen de olika mängderna samt kan associera kunskapen med den som erhölls i animationen på nodnivå. Denna representation gör att användaren lätt kan urskilja att en nod får fler frekvenser ju högre lasten blir. Dessutom skall noden visa hur mycket tid den använder till arbete och hur mycket tid som används till synkronisering, noden visar alltså sin resursutnyttjandegrad. Vi har valt att visa utnyttjandegraden som ett pajdiagram. Pajdiagrammet innehåller tre olikfärgade områden symboliserande den tid som noden är idle, det vill säga den har tillräckligt med frekvenser, den tid som noden väntar på att få välja frekvenser och den tid den

[2] Ett voronoi diagram delar upp ett plan med noder i genom att dra liner på ett sådant sätt att varje punkt på en given linje ligger lika långt ifrån två eller fler noder.



Figur 5.7: En nod i nätverksanimationen av DET-DLC.



Figur 5.8: Ett nätverk i DET-DLC animationen på nätverksnivå.

tillbringar med att välja frekvenser. Dessa olika fält växer och krymper allt eftersom noden tillbringar tid i de olika tillstånden.

Helheten Sammantaget skall det gå att utläsa ur visualiseringen hur belastning gör att frekvenser används av den nod som har störst behov av frekvenser för att utföra sitt arbete. Vi hoppas åstadkomma en gestalteffekt, att användaren skall se de egenskaper som nätverket besitter som inte är lätt härledbart ur de enskilda nodernas beteende. Detta för att studenten skall få förståelse för den dynamik som algoritmen ger frekvensallokeringen. Nätverket av noder har ett beteende som konstitueras av de ingående noderna men det är inte enkelt härledbart ur en nods beteende.

5.3 Prototypens teoriförankring

I detta stycke ska vi återknyta till den teori som vi presenterat tidigare och påvisa hur vi har använt de olika delarna.

Perception och kognition En av de saker som gestaltteorin har bidragit med till vår visualisering är insikten om att det finns en helhet som användaren måste få inblick i. Gestaltteorins principer för gruppering av synintryck är också något som är synligt i vår animation, bland annat hade vi dessa i åtanke då vi genom att ge meddelanden och tillstånds representationerna samma färg och form i Choy-Singh animationen. Vi förlitar oss där på att meddelandena och tillstånden ska grupperas efter denna likhet så att de uppfattas som sammanhängande. Valet att i den första DET-DLC animationen istället använda runda meddelanden gjordes för att frångå denna gruppering.

Att färger bearbetas parallellt är något som utnyttjats flitigt genom att använda mycket färger i animationerna. Vetskapen att det lagras information om hur det går att uttyda information ur grafer i det procedurella minnet låg bakom beslutet att återanvända stora delar av Choy-Singh animationen i DET-DLC animationen på nodnivå.

G A Millers forskning om minnet där han visat att lagringen i korttidsminnet är begränsat är något som hela tiden beaktats genom att vi försökt visa enbart information som är relevant samt att de första två animationerna skall visa enbart ett litet antal noder. I den sista animationen var det dock nödvändigt att visa ett stort antal noder för att det skulle vara möjligt att visa de koncept som vi ville visa och vi minskade där istället den information som visades om varje nod.

Vi har också hela tiden försökt associera det som visas på skärmen med redan kända begrepp för att öka minnesbehållningen hos användaren.

Pedagogisk och övrig forskning De designprinciper som Richard Mayer tagit fram för design av multimediala undervisningshjälpmedel har också hela tiden tagits i beaktande:

1. Multimedia — Trots att vi i denna uppsats inte har gått djupare in påde verbala delarna av animationsdesignen är det tänkt att alla animationer skall användas i samband med en berättare, vilket nämndes i början av designen.

2. Kontinuitet — Detta är något som är starkt sammanknutet med korttidsminnets begränsningar och vi nämnde ovan hur vi tagit hänsyn till detta.

3. Relevans — Vi har i varje steg endast presenterat den information som vi ansett vara nödvändig för att uppnå det syfte vi ställt upp för animationen.

4. Presentationsform — Som nämndes ovan är alla animationer tänkta att presenteras i samband med verbal information.

5. Individuella skillnader — Vi anser att problematiken med individuella skillnader går utanför målsättningen för den här uppsatsen. Vi är medvetna om att den design vi föreslagits skulle vara svår eller till och med omöjlig för till exempel en färgblind person att uppskatta.

I enlighet med Christopher Hundhausens kritik har vi också försökt att *inte* visa en experts mentala bild i våra animationer. I stället har vi försökt visa algoritmen på ett sådant sätt att det grundläggande beteendet först visas för att sedan visa mer avancerade koncept.

Valet att använda mycket färger beror också en del på de studier av färg som informationsförmedlare som gjorts. I enlighet med dessa har vi också noga tänkt igenom vilka färger som skall visas och i vilket syfte. Färg har använts för att markera aktivitet, betona mönster och visa olika tillstånd. Vi har heller inte använt mer än 20-30 färger för att undvika att färganvändningen ger negativt resultat.

5.3.1 Validering av Choy-Singh animationerna

Utifrån de designimplikationer vi ställde upp i teoriavsnittet skall vi här använda dessa för att göra en bedömning av den nuvarande Choy-Singh-animationen och validera den design vi föreslagit.

Kriterier	Nuvarande visualisering	Vårt designförslag
Enkelhet	Icke godkänt	Godkänt
Fullständighet	Icke godkänt	Godkänt
Kausalitet	Icke godkänt	Godkänt
Korrekthet	Icke godkänt	Godkänt
Överblickbarhet	Icke godkänt	Godkänt

Den nuvarande visualiseringen (se sid. 35)

Enkelhet Den nuvarande visualiseringen använder inga intiutiva element i visualisering. Den kräver att studenten läser sig till vad olika färger på meddelanden innebär. Vidare innebär lösningen med pilar som utgår från den nod som sänder meddelandet att pilar läggs ovanpå varandra, vilket gör det svårt/omöjligt att utläsa alla meddelanden som är på väg.

Fullständighet Den nuvarande visualiseringen använder sig av två fönster, ett för att visa nätverket och meddelanden och ett för att visa vilket tillstånd noderna befinner sig i (se sid. 36). Det gör det svårt att bilda sig en uppfattning om hur en nod fungerar, eftersom information om nodens tillstånd och förehavanden är utspritt.

Kausalitet Denna algoritms kausalitet styrs helt av meddelanden. Som nämnts tidigare är meddelanden svåra att uppfatta vilket gör att kausaliteten också blir svår att uppfatta.

Korrekthet Avseende kausaliteten i algoritmen råder en viss diskrepans mellan de två fönstrena avseende noders tillstånd. Som vi tidigare nämnt stämmer inte representation av meddelanden som pilar (se sid. 37) överens med den mentala representation av hur meddelandeskickning som vi antar är allmänt förekommande.

Överblickbarhet Den nuvarande visualiseringen två fönster gör att det krävs många byten av medveten fokus för att samla in den information som skall ligga som grund för en helhetsförståelse av algoritmen. Att det i den nuvarande algoritmen dessutom används ett nätverk med 15 noder gör att sökandet efter den informationen blir komplicerad och tidsödande. Sammantaget ger detta att det är svårt att få en överblick.

Vårt designförslag (se sid. 37)

Enkelhet Vi föreslår en större nod som klarar att visa sitt tillstånd och de möjliga tillstånd den kan befinna sig i samt i vilken ordning noden växlar mellan tillstånd, dessutom finns en textuell representation av tillstånden som ledsagning. Meddelanden har samma färg som det tillstånd det skall beskriva och meddelanden och tillståndsindikatorer har dessutom samma form, vilket underlättar att se samhörigheten mellan tillstånd och meddelanden. Sammantaget blir det enkelt att utläsa information om noderna och nätverket.

Fullständighet Vårt förslag använder bara ett fönster för att visa informationen om algoritmen och nätverket. Det tror vi underlättar förståelsen av hur noder fungerar, som enskilda entiteter och som nätverk.

Kausalitet De meddelanden vi föreslår visar tydligt det kausala samband som finns mellan noders tillstånd samt när och vilket meddelande som skickas vid en given tillståndsförändring.

Korrekthet Eftersom vårt förslag bara använder ett fönster finns det ingen risk att det skall råda diskrepans mellan information avseende noders tillstånd. Diskrepans mellan meddelandeskickning och tillstånd kan undvikas genom en korrekt implementation av visualiseringen.

Meddelanden som objekt som förflyttar sig mellan två noder anser vi vara en korrekt representation av fenomenet meddelande.

Överblickbarhet Vårt förslag till visualisering använder sig av endast sju noder. Det ger ett nätverk som är lätt att överblicka och skapa en helhetsbild utifrån.

6 Diskussion och slutsatser

I hope that posterity will judge me kindly, not only as to the things which I have explained, but also to those which I have intentionally omitted so as to leave to others the pleasure of discovery.
Rene Descartes

6.1 Resultat

Vi har i den här uppsatse designat en prototyp för visualisering av Deterministic Distributed List Coloring. Designen har skett enligt de forskningsresultat som framkommit inom psykologisk och pedagogisk forskning om multimediala undervisningshjälpmedel. Därmed har vi också visat att dessa teorier är applicerbara på design av algoritmanimationer för undervisningssyfte.

6.2 Diskussion

I den här uppsatsen har vi visat att det går att applicera de pedagogiska forskningsresultaten om multimediala undervisningshjälpmedel på design av algoritmanimationer. Detta är något som vi anser bör ge implikationer för utvecklingen av algoritmanimationer med undervisningssyfte.

Att skapa förståelse för en algoritm innebär att bygga associationer mellan händelser och entiteter i algoritmen och händelser och entiteter i arbetsområdet [41]. Därför bör en av de viktigaste frågorna när en algoritm ska visualiseras för att underlätta inlärning av en algoritm vara: Är representationerna i animationen sådana att det verkligen skapas associationer till arbetsområdet?

Att noggrant designa visualiseringen enligt de teorier som vi har presenterat tror vi underlättar att hitta “rätt” representationer. Vi skall nedan presentera ett antal punkter som vi tror är viktiga att tänka på för att åstadkomma en effektiv visualisering.

Representationer

Vi anser att man bör *definiera målgruppen* som animationen skall riktas mot. Genom att identifiera vem animationen riktar sig till får man också reda på vilka kunskaper som finns att bygga vidare på och associera till. Det är viktigt att inte ha representationer som strider mot begreppsliga mentala koder (se till exempel diskussionen om meddelanden sid 38).

Det är också möjligt att den identifierade målgruppen tidigare använt animationer som beskriver liknande algoritmer och det kan då vara fördelaktigt att bygga vidare på de representationer som introducerats i dessa tidigare animationer. Dock måste *representationerna vara anpassade till vad man skall visualisera.* Att bygga vidare på representationer som nästan passar till de koncept man vill visa enbart för att användaren skall känna igen sig förvirrar mer än det hjälper. Det kan då skapas associationer mellan icke relaterade koncept om representationen i animationer är densamma. *Målgruppen ger implikationer för vilka representationer som är lämpliga.*

Vi tror inte heller att det är lyckat att göra animationer med utgångspunkt att de skall ha ett enhetligt utseende. Faran ligger här i att det blir svårt att visualisera de för animationen specifika egenskaperna eftersom representationer som är gemensamma för alla animationer måste användas.

Relevans

Att *definiera ett syfte* för animation är också något vi anser vara viktigt. Beroende på vad som skall visualiseras kommer olika representationer av objekt att krävas samt olika detaljnivå. Det krävs till exempel helt olika representationer för att lära ut ett grundläggande beteende hos en algoritm mot att visa mer övergripande egenskaper hos den. Betrakta till exempel DET-DLC animationen på nodnivå kontra nätverksnivå (se sidorna 40 – 43). *Syftet med animationen definierar vilka av algoritmens egenskaper som är relevanta.*

Kontinuitet

När syftet definieras bör en avgränsning göras så att det inte förutsätts att en animation skall demonstrera ett stort antal koncept. Hänsyn måste tas till den begränsade kapaciteten hos användarens korttidsminne, den kognitiva lasten får inte bli för stor eftersom det då blir svårt att uppnå kontinuitet i animationen. Vi anser att det är bättre att dela upp animationen i ett antal delanimationer, var och en med ett väl avgränsat syfte för att minska den information som måste presenteras samtidigt.

Att dela upp en animation i flera fönster med avsikt att på detta sätt kunna presentera mer information är något som vi inte heller tror på i en undervisningssituation. Då det används ett flertal fönster för att visa en algoritm måste användaren hela tiden skifta sin uppmärksamhet mellan de olika fönstrena. Detta ökar den kognitiva lasten och gör det svårare att uppnå kontinuitet i animationen. Dock kan multipla fönster passa alldeles utmärkt i animationer med andra syften än just inlärning. *Kontinuiteten begränsar syftets omfattning.*

6.3 Fortsatt forskning

Den naturliga fortsättningen på detta arbete är att implementera och evaluera visualiseringen enligt designförslagen. Det är viktigt att innan en evaluering görs även designa den verbala delen grundligt.

Utöver detta skulle det också vara intressant att undersöka effekten av att använda olika grafiska representationstekniker för att beskriva konceptuella begrepp i algoritmer.

Litteraturförteckning

[1] Allwood, Carl M. *Människa-datorinteraktion, Ett psykologiskt perspektiv*. Lund, Studentlitteratur, 1998

[2] Badre, Albert, *et al*. "Assessing Program Visualization Systems as Instructional Aids" Technical Report GIT-GVU-91-23, GVU Center, Georgia Institute of Technology, Atlanta, USA, okt 1991

[3] Badre, Albert, Clayton Lewis, John T. Stasko. "Do Algorithm Animations Assist Learning? An Empirical Study and Analysis" i *Proceeding of the INTERCHI '93 Conference on Human Factors in Computing Systems*, Amsterdam, Netherlands, apr 1993

[4] Badre, Albert, Andrea W. Lawrence, John T. Stasko. "Empirically Evaluating the Use of Animations to Teach Algorithms" I *Proceedings of the 1994 IEEE Symposium on Visual Languages*, 48-54, St. Louis, USA, 1994

[5] Baecker Ronald. "Sorting out Sorting: A Case Study of Software Visualization for Teaching Computer Science" ur *Software Visualization, Programming as a Multimedia Experience*, Cambridge, Massachusetts, USA, The MIT Press, 1998

[6] Bergin, Joe *et. al.* "An overview of visualization: its use and design. Report of the Working Group on Visualization" Barcelona, Spanien, 1996

[7] Brown, Marc H., John Hershberger. "Color and Sound in Algorithm Animation" IEEE Computer, Vol. 25, No. 12, 52-63, 1991

[8] Byrne, Michael D., Richard Catrambone, John T. Stasko. "Do Algorithm Animation Aid Learning." Technical Report GIT-GVU-96-18, GVU Center, Georgia Institute of Technology, Atlanta, GA, USA, aug 1996

[9] Choy, Manhoi, Ambuj K. Singh. "Efficient Fault-Tolerant Algorithms for Distributed Resource Allocation" *ACM TOPLAS*, Vol 17, No 3, pp. 535-559, maj 1995

[10] Corbett, Albert T., Bonnie E. John och John F. Pane. "Assessing Dynamics in Computer-based Instruction" I *Proceedings of the ACM SIGCHI ´96 Conference on Human Factors in Computing System*, 197-204, Vancouver, B.C., Canada, apr 1996

[11] Ebbinghaus, Hermann. *Memory: A Contribution to Experimental Psychology*. 1885
[URL: http://www.yorku.ca/dept/psych/classics/Ebbinghaus/index.htm]

[12] Encyclopedia Brittanica Online, [URL: http://www.eb.com:180]

[13] Elkerton, Jay, Susan Palmiter. "An Evaluation of Animated Demonstrations for Learning Computer-based Tasks" I *Proceedings of the ACM SIGCHI ´91 Conference on Human Factors in Computing Systems*, 257-263, New Orleans, LA, USA, maj 1991

[14] Garg, Naveen, Marina Papatriantafilou och Philippas Tsigas. "Distributed List Coloring: How to Dynamically Allocate Frequencies to Mobile Base Stations"

[15] Gordon, Ian E. *Theories of Visual Perception.* New York, USA, John Wiley & Sons, 1997

[16] Harp, Shannon F., Richard E. Mayer. "Role of interest in learning from scientific text and illustrations: On the distinction between emotional interest and cognitive interest" *Journal of Educational Psychology*, Vol 89, 92-102, 1997

[17] Harp, Shannon F., Richard E. Mayer. "How seductive details do their damage: A theory of cognitive interest in science learning" *Journal of Educational Psychology*, Vol. 90, 414-434, 1998

[18] Hoadley Ellen D., "Investigating the Effects of Color" *Communications of the ACM*, Vol. 33, No. 2, 1990

[19] Hundhausen, Cristopher D. *Toward Effective Algorithm Visualization Artifacts: Designing for Participation and Communication in an Undergraduate Algorithms Course* URL:http://www.hawai.com, jun 1999

[20] James, William. *The Principles of Psychology*, 1890
[URL: http://www.yorku.ca/dept/psych/classics/James/Principles/index.htm]

[21] Kehoe, Colleen M., John T. Stasko. "Using Animations to Learn about Algorithms: An Ethnographic Case Study" Technical Report GIT-GVU-96-20, GVU Center, Georgia Institute of Technology, Atlanta, GA, USA, sep 1996

[22] Kehoe, Colleen M., John T. Stasko och Ashley Taylor. "Rethinking the Evaluation of Algorithm Animations as Learning Aids: An Observational Study" Technical Report GIT-GVU-99-10, GVU Center, Georgia Institute of Technology, Atlanta, GA, USA, mar1999

[23] Kraemer, Eileen. "Visualizing Concurrent Programs" *Software Visualization, Programming as a Multimedia Experience*, Cambridge, Massachusetts, USA, The MIT Press, 1998

[24] Koldehofe, Boris, Marina Papatriantafilou och Philippas Tsigas. “Distributed Algorithms Visualisation for Educational Purposes.” I *Proceedings of the ITiCSE ’99* (pp. 103-106), Krakow, Polen, jun 1999

[25] Köhler, Wolfgang. “Gestalt Psychology Today” *American Psychologist*, Vol. 14, 727-734, 1959
[URL: http://www.yorku.ca/dept/psych/classics/Kohler/today.htm]

[26] Lohse, Jerry. “A Cognitive Model for the Perception and Understanding of Graphs”. I *Human factors in computing systems conference proceedings on Reaching through technology*, 137-144, 1991

[27] Lundh, Lars-Gunnar, Henry Montgomery, Yvonne Waern. *Kognitiv psykoklogi*. Lund, Studentlitteratur, 1992

[28] Mayer, Richard E., Richard B. Anderson. “Animations Need Narrations: An Experimental Test of Dual-Coding Hypothesis” *Journal of Educational Psychology*, Vol 83. No. 4, 484-490, 1991

[29] Mayer, Richard E., Richard B. Anderson. “The Instructive Animation: Helping Students Build Connections Between Words and Pictures in Multimedia Learning” *Journal of Educational Psychology*, Vol 84. No. 4, 444-452, 1992

[30] Mayer, Richard E., Valerie K. Sims. “For Whom Is a Picture Worth a Thousand Words? Extensions of a Dual-Coding Theory of Multimedia Learning” *Journal of Educational Psychology*, Vol 86. No. 3, 389-401, 1994

[31] Mayer, Richard E. *et al.* “When less is more: Meaningful learning from visual and verbal summaries of science textbook lessons” *Journal of Educational Psychology*, Vol 88. 64-73, 1996

[32] Mayer, Richard E. “Multimedia Learning: Are We Asking the Right Questions?” *Educational Psychologist*, Vol 32. No. 1, 1-19, 1997

[33] Mayer, Richard E., Moreno, Roxana. “A split-attention effect in multimedia learning: Evidence for dual information processing systems in working memory” *Journal of Educational Psychology*, Vol 90, 312-320, 1998

[34] Mayer, Richard E. “Research-based principles for the design of instructional messages” *Document Design*, Vol 1. No. 1, 7-20, 1999

[35] Mayer, Richard E. *et al.* “Maximizing Contructivist Learning From Multimedia Communications by Minimizing Cognitive Load” *Journal of Educational Psychology*, Vol. 91, No. 4, 638-643, 1999

[36] Miller, George A. “The Magical Number Seven, Plus or Minus Two: Some Limits on Our Capacity of Processing Information.” *The Psychological Review*, No. 63, 81-97, 1956

[37] Moreno, Roxana, Richard E. Mayer. “Cognitive Principles of Multimedia Learning: The Role of Modality and Contiguity” *Journal of Educational Psychology*, Vol. 91, No. 2, 358-368, 1999

[38] Moses, Yoram, Zvi Polunsky, Ayellet Tal, Leonid Ulitsky. “Algorithm Visualization For Distributed Environments.” *IEEE Symposium on Information Visualization ’98.* 1998

[39] Paivio, Allan. *Imagery and Verbal Processes.* New Jersey, USA, Lawrence Erlbaum Associates, 1979

[40] Paivio, Allan. *Mental Representations.* New York, USA, Oxford University Press, 1986

[41] Petre, Marian, Alan Blackwell och Thomas Green. “Cognitive Questions in Software Visualization” ur *Software Visualization, Programming as a Multimedia Experience,* edited by John Stasko *et al*. Cambridge, Massachusetts, USA, The MIT Press, 1998

[42] Tel, Gerard. *Introduction to Distributed Algoritms*. New York, USA, Cambridge University Press, 1994

[43] Tufte, Edward R., *Envisioning Information.* Cheshire, Connecticut, USA, Graphics Press, 1990

[44] Wertheimer, Max. “Gestalt theory”, 1924
[URL: http://www.enabling.org/ia/gestalt/gerhards/wert1.html]

För övrigt anser vi att Redmond bör förstöras.