CHALMERS

Okonventionell beräkning

En introduktion

Elias Hakuni
László Sall Vesselényi
Viktor Blomqvist

Handledare: Zoran Konkoli
Examinator: Vessen Vassilev

Kandidatarbete MCCX02-15-4
Institutionen för mikroteknologi och nanovetenskap
Chalmers Tekniska Högskola
Göteborg, Sverige 2015

Sammanfattning

Detta litteraturbaserade arbete avser att ge en introduktion till fältet okonventionell beräkning. Okonventionell beräkning är ett begrepp och forskningsfält i gränslandet mellan matematik, datavetenskap, informationsteknik, filosofi och naturvetenskap. För arbetets syfte behöver det konventionella beräkningsgreppet först redogöras för. Konventionell beräkning kan ses som bestånde av två grenar: Dels en gren som utgörs av konventionella beräknare, den moderna digitala datorn; dels en som utgörs av konventionella beräkningsmodeller, turingmaskinen och Church-Turings hypotes. Turingmaskinen, utvecklad av Alan Turing på 1930-talet, är en formalisering av algoritmer. Church-Turings hypotes hävdar helt enkelt att turingmaskinen, och alla modeller ekvivalenta med densamma, kan beräkna alla funktioner som är beräkningsbara av en människa med penna och papper följandes en algoritm. Bakgrunden till dessa två resultat är den stora frågan: ”Vad går att beräkna?” Bryggan från konventionell beräkning till okonventionell beräkning är en breddning av beräkningsgreppet: en process som övergår från ett ursprungstillstånd till ett sluttillstånd. Med denna bredare definition öppnas ett större fält av beräkningar, beräknare och beräkningsmodeller än vad som ryms inom den rådande konventioner. Denna definition innefattar inte längre bara beteendet hos en digital dator eller en människa med papper och penna utan potentiellt allt från kvantfysiska fenomen och kemiska reaktioner till betraktelsen av en sten över tid och universums samlade företeelser. Implicit eller explicit är denna breda definition central för okonventionell beräkning, och viktiga frågor blir därför: ”Vad kan beräkna?”; ”Vilka kriterier behövs för en beräknare?”; och en återkoppling till den konventionella frågan: ”Vad kan beräknas?” Introduktionen av okonventionell beräkning slutförs genom exempel på tre olika okonventionella beräknare och beräkningsmodeller: Artificella neuronnät, kvantberäknare och reaktions-diffusionsberäknare. Beräknarnas och beräkningsmodellernas egenskaper, svagheter och eventuella användnings- och tillämpningsområden presenteras, såväl som hur de anses vara okonventionella.

Abstract

This thesis, based on literature studies, aims to provide an introduction to the field of unconventional computation. Unconventional computation is a concept and field of research inhabiting the intersection between mathematics, computer science, information technology, philosophy and the natural sciences. For the purpose of the thesis, conventional computation must first be accounted for. Conventional computation can be seen as consisting of two branches: a branch for conventional computers, the modern digital computer; another for conventional models of computation, the Turing machine and the Church-Turing thesis. The Turing machine, developed by Alan Turing on the 1930's, is a formalisation of algorithms. The Church-Turing thesis posits simply that the Turing machine, and all models equivalent with it, can compute all functions computable by a human with pen and paper following an algorithm. In the background of these two results is the big question: "What can be computed?" Bridging from conventional computation to unconventional computation is a widening of the concept of computation: A process that transitions from a start state to an end state. With this wider definition, a larger field of computations, computers and models of computations opens up than is contained within the predominant conventions. This definition does no longer only contain the behaviour of a digital computer or a human with pen and paper but potentially everything from quantum physical phenomena and physical reactions to the observation of a rock over time and the collected events of the universe. Implicitly or explicitly, this wide definition is central to unconventional computation, and important questions are thus: "What can compute?"; "What criteria are required for a computer?"; and a return to the conventional question: "What can be computed?" The introduction of unconventional computation is completed through examples of three different unconventional computers and models of computation: Artificial neural networks, quantum computers and reaction-diffusion computers. The features, weaknesses and possible fields of applications of the computers and models of computation are presented, as well as how they are considered to be unconventional.

Förord

Författarna till denna rapport vill tacka sin handledare, Zoran Konkoli, för all hjälp och vägledning denna våren. Tack till de kandidatgrupper som läst och kommenterat texten innan den var färdig. Tack till biblioteket som tillhandahållit informationskompetenskurs och alla de artiklar denna studie baserats på. Tack fackspråk för handledning och föreläsningar.

Innehåll

Kapitel 1

Inledning

Beräkningar är idag, genom datorer och andra besläktade tekniska system, en fundamental del av samhället. Trots detta är variationen i hur beräknare konstrueras och hur beräkningar formuleras förhållandevis liten. Okonventionell beräkning handlar om att undersöka andra sätt att formulera och implementera beräkning.

1.1 Vad är en beräkning?

För att kunna diskutera hur beräkningar är konventionella eller okonventionella behöver först begreppet beräkning definieras. Beräkning är starkt förknippat med de numeriska beräkningar som utförs av datorer. I denna studie ses dock beräkning som något mera allmänt. Först och främst behövs en beräknare som utför beräkningen (som dock inte behöver vara konkret). Beräkning kan utifrån detta beskrivas som den serie förändringar beräknaren genomgår när den börjat vid något ursprungstillstånd och hamnar vid något sluttillstånd (Siegelmann, 1999, s.147). Detta i sig skulle betyda att alla system som genomgår förändringar också därmed genomför någon beräkning. Om detta är fallet och i så fall exakt vilka system som därmed är beräknare är en icke-trivial fråga. Till exempel har det debatterats huruvida en sten utför beräkningar, och i så fall vilka (Chalmers, 1996). Ibland beskrivs tillståndet hos ett beräknande system som informationen det innehåller. Beroende på hur ett systems tillstånd tolkas kan olika information representeras. Olika tolkningar kan betyda att samma tillstånd utgör olika information (Piccinini, 2012).

I ett uppenbart beräknande system, en räknande människa med penna och papper, är start- och ursprungstillstånden kombinationer av symboler som kan finnas på pappret. Förändringarna som systemet genomgår är i detta fall de förändringar som sker då människan skriver eller suddar på pappret. Varje steg i förändringen är en diskret operation som potentiellt föregås och åtföljs av andra liknande steg, och om det människan beräknar har en lösning leder dessa steg till slut fram till lösningen.

Den moderna konventionella synen på beräkningar och beräkningsteori har sin grund i arbetet som Alan Turing, Alonzo Church och andra forskare påbörjade under 1930-talet (Davis, 2006). Deras forskning omvandlade begreppet beräkning från att vara en löst definierad process i fysiska system, speciellt människor som beräknar med penna och papper, till formella processer som lyder under väldefinierade regler. Turing med sina turingmaskiner och Church med λ-kalkylen gav formella verktyg för att resonera kring algoritmer, som ansågs beskriva beräkningar (Fernandez, 2009).

1.2 Konventionell beräkning

En fysisk konventionell beräknare är inte särskilt oväntat det som i vardagligt tal går under namnet dator. I en dator samspelar minne av olika slag med en processor som tar emot och skickar iväg elektriska signaler. Processorer utgörs enkelt förklarat av ett mycket stort antal transistorer på ett chip. Transistorerna är sammankopplade på ett sådant vis att de bildar logiska portar vilka gör det möjligt att manipulera de elektriska signalerna

som når processorn (Hodges och (e-book collection), 2007). Då transistorerna i regel består av halvledaren kisel benämns konventionell beräkning ibland kiselbaserad beräkning. Utvecklingen av kiselbaserade processorer har under flera årtionden huvudsakligen följt en exponentiell tillväxt, benämnd Moores lag och formulerad av Mack (2006) som: Antalet transistorer som går att placera på en given yta kommer att fördubblas vartannat år.

Konventionerna kring synen på vad en beräkning är bygger på den formalisering av beräkningar som skedde under 30-talet. I både Turings och Churchs modeller uttrycks beräkningar som något som utförs algoritmiskt, och denna uppfattning, Church-Turings hypotes, har inte motbevisats sedan dess formulering (Moore och Mertens, 2011). Olika konventionella sätt att formulera beräkningar har inte utökat den grundläggande principen om algoritmisk beräkning men väl bidragit till olika problemlösningsparadigm inom konventionen (Fernandez, 2009).

1.3 Okonventionell beräkning

Okonventionella beräkning kan ta sig uttryck i såväl abstrakta modeller som fysiska implementationer. Gemensamt för båda kategorier är att någon eller några av de konventioner som råder inom beräkningsområdet har brutits. De fysiska okonventionella beräknarna kan vara implementationer av okonventionella koncept eller arkitekturer. De kan även följa den logikportsbaserade arkitekturen som konventionella datorer nyttjar men vara konstruerade av andra material än halvledare (Adamatzky, 2014).

Dock behöver en okonventionell beräkning, precis som för beräkningar i allmänhet, inte ske i ett fysiskt system. Genom att utgå från okonventionella arkitekturer och koncept för teoretiska resonemang kring beräkningar, som därigenom kan betraktas som okonventionella, kan nya insikter vinnas (Adamatzky, 2014). Detta kan, likt variationer av den konventionella förståelsen för beräkningar, bidra till nya, specialanpassade sätt att formulera och lösa problem på (Harel, 2000).

1.4 Syfte

Syftet med denna studie är att ge en introduktion till okonventionell beräkning. Därmed syftar studien till att ge exempel av vad okonventionell beräkning är, hur en okonventionell beräknare kan fungera och vad den kan användas till. För att på ett meningsfullt sätt kunna diskutera okonventionell beräkning innehåller studien även en introduktion till konventionell beräkningsteori och hur denna kan användas vid resonemang kring beräkning.

1.5 Avgränsningar

Innehållet i denna rapport är uteslutande ett resultat av litteraturstudier och inga implementationer eller praktiska experiment med okonventionella beräknare har genomförts. Studien har inte syftat till att ge en inblick i alla kategorier av okonventionella beräknare utan har begränsat sig till tre stycken valda okonventionella beräknare och beräkningsmodeller.

1.6 Översikt

Kapitel 1 syftar till att ge en introduktion till vad okonventionell beräkning är, varför det är relevant samt förklara hur vissa begrepp tolkas inom detta område. Förutom detta

presenteras också projektets syfte och en överblick ges över studiens omfång.
Kapitel 2 beskriver arbetets metod.
I **kapitel 3** redogör för formaliseringen av beräkningsbegreppet i turingmaskinen och delar av beräkningsteorin. Utifrån denna teoretiska grund presenteras Church-Turings hypotes och tolkningar av denna. Frågeställningar kring beräkning och fysiska beräknare som är relaterade till beräkningsteorin tas också upp.
Kapitel 4 inleds med en presentation av okonventionell beräkning som koncept och identifierar vilka frågor kring en beräknare som är av intresse. Efter introduktionen följer avsnitt om tre olika okonventionella beräknare. Hur de fungerar, vad de kan användas till, vad de beräknar, är några av frågeställningarna.
Den avslutande diskussionen i **kapitel 5** reflekterar över de resultat presenterats i föregående kapitel.

Kapitel 2

Metod

Arbetet som denna rapport baserats på har främst bestått av litteraturstudier kring okonventionella beräknare samt närliggande relevanta områden. I början av projektet låg fokus i studierna på att introducera projektgruppen själva till ämnet då det till stor del låg utanför gruppens förkunskaper. Efterhand som gruppen fått ett fastare grepp om vilka delar som är centrala för syftet har fokus i litteraturstudierna skiftat till att handla om insamling av källor, referenser och citat som kan understödja det som presenteras i denna rapport.

I regel har arbetet med att söka upp och bedöma litteratur delats upp mellan gruppens medlemmar, oftast med syftet att besvara någon fråga eller undersöka något område. Därefter har det material som bedömts relevant och lovande diskuterats inom gruppen som gemensamt kommit fram till vad som bör inkluderas i rapporten och om något är extra viktigt.

De avsnitt i 4 och 5 där tre olika okonventionella beräknare presenteras och diskuteras har producerats på annorlunda vis. Varje gruppmedlem har valt en okonventionell beräknare att fördjupa sig i och skriva om. Valet av beräknare motiverades dels utifrån gruppens tycke men tog också hänsyn till en viss spridning hos områdena som beräknarna tangerar.

Kapitel 3

Konventionella beräkningsmodeller

För att på ett allmänt sätt kunna resonera kring just beräkningar utan hänsyn till en enskild beräknares egenskaper krävs en abstraktion från beräknarna, en beräkningsmodell (Fernandez, 2009). Beräkningsmodeller är således abstraherade formaliseringar av beräkningsbegreppet som tillåter resonemang kring vad beräkningar är, vad som går att beräkna (och hur) och vad som kan beräkna. Beräkningsmodellers huvudsakliga användningsområde har framför allt varit att fastställa vad en beräkning är och utifrån det utforska och försöka besvara frågor om vad som därmed går att beräkna (Sipser, 2012). Sedermera har detta utvecklats till två distinkta men samverkande forskningsfält: beräkningsteori, vad går att beräkna, och komplexitetsteori, hur (resurseffektivt) går det att beräkna (Fernandez, 2009). Detta verk, och detta kapitel i synnerhet, har sitt fokus på beräkningsteori.

Bakgrunden till de konventionella beräkningsmodellernas utveckling är att det under decennierna efter sekelskiftet bedrevs intensivt vetenskapligt arbete på frågor om matematikens grunder (Piccinini, 2012; Kozen, 1997). En central fråga var vad som går att beräkna, med avseende på den fram till dess dominerande beräknaren: människan med penna, papper och regler (Copeland, 2008). I försök att besvara denna fråga formulerades olika problem under denna period, såsom avgörandeproblemet[1] (Fernandez, 2009). Avgörandeproblemet frågar helt enkelt om det för något godtyckligt logiskt systemet och ett godtyckligt påstående i system finns en algoritm som avgör om påståendet är sant eller falskt. Alan Turing var en av de som under 30-talet fann att det finns en klass av problem, till vilken avgörandeproblemet hör, som inte är beräkningsbara. Detta gjorde dessa matematiker just medelst beräkningsmodeller, med vilka de abstraherade och formaliserade den mänskliga beräkningen. Dessa beräkningsmodeller utgör den gemensamma grunden för den moderna beräkningsteorin och kallas därmed de konventionella beräkningsmodellerna (Moore och Mertens, 2011).

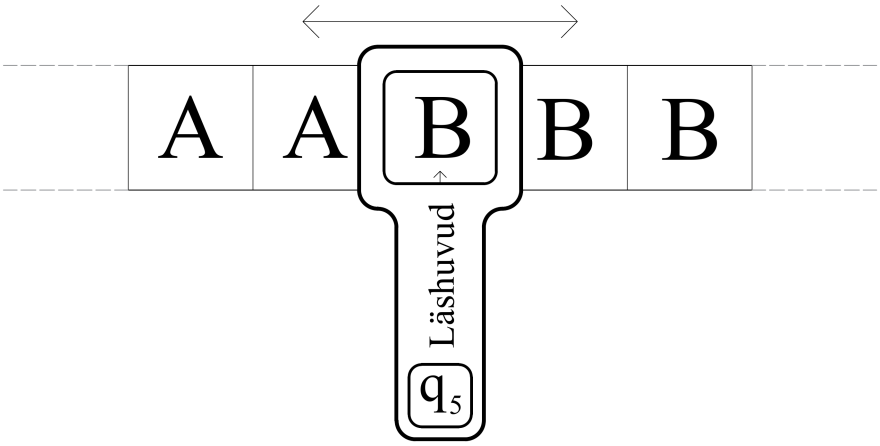
Detta kapitel börjar med en beskrivning och förklaring av Turings beräkningsmodell, turingmaskinen, hur den fungerar och dess betydelse för datavetenskapen. Därefter berörs några andra konventionella beräkningsmodeller och deras relation till turingmaskinen kort, för att sedan övergå till automater, med vars hjälp det går att modellera olika nivåer av beräkningskraft. Efter en beskrivning av beräkningskraftsnivån som kallas super- och hyperturing avslutas kapitlet därefter med en genomgång av Church-Turingtesen och för arbetet relevanta tolkningar av densamma.

3.1 Turingmaskinen

Turingmaskinen (eller *"a-machine"* från "automatic machine", som Turing (1937) själv kallade den) är en hypotetisk maskin, utvecklad av Alan Turing, som består av ett läshuvud samt ett band av obegränsad men ändlig längd som är delat i diskreta rutor (Hopcroft,

[1] Från tyskans "Entscheidungsproblem", formulerat av matematikern David Hilbert

Motwani och Ullman, 2006). Detta band, som innehåller ett ändligt antal symboler, kan läshuvudet både läsa ifrån och skriva till. Figur 3.1 visar ett sätt att illustrera läshuvudet och bandet. De symboler som en turingmaskins läshuvud hanterar är maskinens alfabet Σ (ibland uppdelat i indataalfabet Σ och bandalfabet Γ, där $\Sigma \subseteq \Gamma$ och Γ innehåller den blanka symbolen $\square$ som omger indata). Läshuvudet har en ändlig mängd tillstånd $Q = \{q_0,...,q_n\}$ det kan befinna sig i och det kan röra sig längs bandet en ruta i taget åt vänster eller höger, ofta betecknat L respektive R. Därtill har varje turingmaskin en uppsättning övergångsfunktioner $\delta : Q \times \Sigma \to Q \times \Sigma \times \{L, R\}$, som med huvudets tillstånd och den inlästa symbolen bestämmer huvudets beteende: vilket tillstånd huvudet byter till, vilken symbol huvudet skriver till rutan det precis läste av och vilken riktning huvudet tar ett steg i (Hopcroft et al., 2006).



Figur 3.1: En möjlig visualisering av en turingmaskin med läshuvud och band. På bandet finns ett antal symboler skrivna som läshuvudet kan läsa av. Kombinationen av läshuvudets tillstånd, här q_5, och den för läshuvudet synliga symbolen, här B, avgör vilket av maskinens alla tillstånd Q läshuvudet övergår till, vilken symbol ur maskinens alfabet Σ läshuvudet skriver till rutan den står ovanför och vilken riktning, höger eller vänster, läshuvuduet tar ett steg i.

Sättet på vilket en turingmaskin utför en beräkning är att läshuvudet, med start i ursprungstillståndet q_0 och på symbolen längst till vänster, läser symbolerna på bandet, byter tillstånd, skriver symboler och förflyttar sig längs bandet; läshuvudet följer sina övergångsfunktioner δ. Läshuvudet följer övergångsfunktionerna δ ända tills det terminerar i och med att det saknas en övergångsfunktion δ för en viss tillstånd-symbolkombination eller hamnar i ett av sluttillstånden F. Det senare används för att underlätta urskiljandet mellan en framgångsrik och en misslyckad eller felaktigt avbruten beräkning: En framgångsrik beräkning slutar alltid i något av tillstånden i F, och då sägs turingmaskinen ha accepterat symbolkombinationen den matades med. Utöver accepterandet eller avfärdandet av den inmatade symbolkombinationen kan även den färdigmanipulerade symbolkombination ses som turingmaskinens utdata (Hopcroft et al., 2006; Fernandez, 2009).

En speciell kategori av turingmaskiner är *universella turingmaskiner* som kan simulera andra turingmaskiner (Turing, 1937). En universell turingmaskin U gör detta genom få tillstånden Q_M, sluttillstånden F_M och övergångsfunktionerna δ_M, kodade på ett lämpligt vis (vanligtvis som binära strängar), för en annan turingmaskin M som indata tillsammans med den egna indatan w för den simulerade turingmaskinen M (Fernandez, 2009). Därefter utför den universella turingmaskinen U en beräkning på hela indatan Mw. En enda universell turingmaskin U kan därmed utföra godtyckliga beräkningsbara beräkningar (där andra turingmaskiner behöver konstrueras specifikt för den önskade beräkningen)

(Fernandez, 2009).

Inom litteraturen om turingmaskiner finns flera varianter på turingmaskinen, bland annat icke-deterministiska turingmaskiner och turingmaskiner med multipla band (Sipser, 2012). Alla dessa har dock samma beräkningskraft som den enkla, deterministiska turingmaskinen och behandlas inte vidare detta verk.

3.1.1 Turingmaskinens tillämpning, algoritmer och digitala datorer

Mer än bara beräkningsbegreppet formaliserade Turing i turingmaskinen även algoritmer, ändliga processer som med entydiga regler och ett begränsat antal diskreta steg löser ett problem (Fernandez, 2009). Detta ansågs beskriva den mänskliga beräkningsförmågan, vad som kallas intuitiv (Piccinini, 2011) eller effektiv (Cotogno, 2003) beräkning, och utgör också grunden för de moderna konventionella beräknarna, de digitala datorerna. Således automatiserades den mänskliga beräkningsprocessen genom implementation av turingmaskinen i digitala datorer (Moore och Mertens, 2011).

Hos digitala datorer motsvarar processorn turingmaskinens läshuvud och arbetsminnet maskinens band (Fernandez, 2009). Digitala datorer motsvarar i sin helhet inte heller vilken turingmaskin som helst utan universella turingmaskiner; precis som en universell turingmaskin U läser den in en annan turingmaskin M (ett program, ett operativsystem etc.) och exekverar det med eventuellt någon mer indata w (som eventuellt även kan vara tom). En viktig egenskap hos turingmaskinen är avsaknaden av någon begränsning i minneskapacitet, men i praktiken går många intressanta beräkningar att genomföra med datorns begränsade mängd minne (och hypotetiskt kan begränsningen i viss utsträckning åtgärdas av hårddiskar och externa minnestillbehör) (Piccinini, 2012; Fernandez, 2009). Dessutom kan en dator, just i och med dess grund i turingmaskinen och algoritmisk beräkning, endast behandla en begränsad mängd data givet en begränsad tidsåtgång.

Ett viktigt exempel på användning av turingmaskiner, och besläktat med avgörandeproblemet, är det så kallade *stopproblemet*[2]: går det att beräkna ifall en godtycklig turingmaskin med godtycklig indata kommer att terminera? Det är bevisat av Turing och hans kollega Alonzo Church att det inte är möjligt (Fernandez, 2009). Innebörden för digitala datorer är att det är omöjligt att skriva ett program som testar ifall ett godtyckligt program med godtycklig indata kommer att terminera framgångsrikt (Mitchell, 2003).

3.2 Andra konventionella beräkningsmodeller

Turingmaskinen är inte det enda försöket att formalisera beräkningsbegreppet för att kunna resonera kring dess egenskaper och begränsningar. Ett flertal andra beräkningsmodeller har utvecklats såväl innan som efter turingmaskinen, varav två samtida med just turingmaskinen är λ-kalkylen utvecklad av Alonzo Church och generella rekursiva funktioner som formulerad av Kurt Gödel och Stephen Kleene (Fernandez, 2009). Vad som är essentiellt med dessa två är att de båda är: Turingkompletta, vilket innebär att de kan simulera alla turingmaskiner och därmed beräkna alla funktioner en turingmaskin kan; och ekvivalenta med turingmaskinen, vilket betyder att de kan simuleras av någon turingmaskin och funktionerna de kan beräkna kan beräknas av någon turingmaskin (distinktionen mellan turingkomplett och ekvivalens med turingmaskinen berörs åter i avsnitt 3.4). En viktig konsekvens av detta är att det för en annan beräkningsmodell, ett programmeringsspråk eller något annat system räcker att visa att systemet är ekvivalent med någon av de tre

[2]Från engelskans ”Halting problem”.

nämnda beräkningsmodellerna för att visa att det kan beräkna alla beräkningsbara funktioner (Fernandez, 2009). Mer om turingmaskiner och de beräkningsbara funktionerna, uttryckt i Church-Turingtesen, skrivs i avsnitt 3.5.

3.3 Beräkningskraft och automater

De tre konventionella beräkningsmodellerna, är ekvivalenta, men för den okonventionella beräkningen är det även viktigt hur de är ekvivalenta, och vad det innebär att inte vara ekvivalent med turingmaskinen. För att reda ut dessa frågor görs en genomgång av begreppet beräkningskraft, såväl som familjen av beräkningsmodeller som kallas automater.

3.3.1 Beräkningskraft

Beräkningskraft betecknar vad, i termer av exempelvis funktioner, problem eller tal, en given beräkningsmodell eller beräknare klarar av att beräkna, oberoende hur kostnadseffektivt med avseende på exempelvis tid eller minne detta sker (Fernandez, 2009). Exempelvis har alla konventionella digitala datorer samma beräkningskraft (förutsatt att de inte begränsas av mängden minne, mer om detta i avsnitt 3.1.1) som de konventionella beräkningsmodellerna, däribland turingmaskinen, då de kan genomföra samma beräkningar, givet tillräckligt med tid och minne och rätt programvara (Piccinini, 2012; Fernandez, 2009).

Begreppet beräkningskraft ska särskiljas från hur resurseffektivt en beräknare eller en modell kan genomföra en given beräkning. Inom litteraturen görs ibland åtskillnad mellan ”computational power” eller ”expressive power” å ena sidan och ”computing power” å andra (Harel och Feldman, 2004; Mitchell, 2003). De förra betecknar det begrepp som definierats i stycket ovan och det senare hur exempelvis snabbt en beräkning kan genomföras. För ”computing power”, som kan översättas till prestanda eller kapacitet, är sådana egenskaper som ekvivalens med turingmaskinen och turingkompletthet betydelselösa, då de endast säger vad en beräkningsmodell eller beräknare kan beräkna, inte hur: två realiserade turingmaskiner, varav en tar ett steg per sekund och den andra en miljard steg per sekund, har båda samma beräkningskraft men olika prestanda (Mitchell, 2003). Då olika beräkningsmodeller och beräknare kan vara avsedda att tillämpas på en viss klass av problem eller optimerad för en viss parameter (som bland annat tid eller minne) och för olika storleksordnignar kan det dessutom vara problematiskt att jämföra prestandan mellan olika system eller modeller (Koffman och Wolfgang, 2010).

3.3.2 Automater

Automater är i sin grund modeller för diskreta processer och är alltså även de beräkningsmodeller (Hopcroft et al., 2006). Det särskilda med automater är att det är en bred klass med modeller som kan användas till att uttrycka olika nivåer av beräkningskraft. Därtill är automater, oavsett nivå på beräkningskraft, i allmänhet simpla i det att den beräkning de utför är beslutsberäkningar: beräkningen slutar i ett ”ja” eller ”nej”. Vad som ligger bakom detta är att automater åtminstone består av en uppsättning tillstånd Q, varav ett grundtillstånd q_0 och ett eller flera sluttillstånd F, en mängd symboler i ett alfabet Σ som olika kombinationer fungerar som automatens indata och ett antal övergångsregler $\delta : Q \times \Sigma \to Q$. Vissa automater har tillägg till detta, som exempelvis grundläggande minnesfunktioner, och vid noga anblick påminner automaters grundläggande beståndsdelar om turingmaskinens (som de presenterades i början på avsnitt 3.1) (Fernandez, 2009).

Turingmaskinen benämns också som en automat och med sin beräkningskraft utgör den således den mest kraftfulla av automater, framför allt tack vare dess obegränsade minne (det går att konstruera annorlunda automater med ekvivalent beräkningskraft, men deras detaljer är inte av intresse för detta verk) (Fernandez, 2009). De övriga automaternas roll i detta är alltså att olika automater, beroende på hur avancerade koncept de implementerar och hur avancerad deras struktur är, formaliserar olika beräkningskraftsnivåer, vilket tar sig uttryck i hur komplicerade mönster på indatan de känner igen (Fernandez, 2009).

3.4 Superturing och hyperturing

Turing själv visade redan kort efter publiceringen av sitt arbete med turingmaskinen möjligheten att formulera turingkompletta modeller med större beräkningskraft än turingmaskinen (Turing, 1939), och i litteraturen finns det framför allt två begrepp för beräkningsmodeller och beräknare som överstiger turingmaskinens kraft: Superturing och hyperturing. Superturing (Siegelmann, 1995) betecknar beräkningsmodeller som kan beräkna problem som ingen turingmaskin kan och som föreslås kunna konstrueras, realiseras i en fysisk beräknare. Likaså betecknar hyperturing (Copeland och Proudfoot, 1999; Cotogno, 2003) beräkningsmodeller som överstiger turingmaskinens kraft, men med skillnaden att inga realiseringar har föreslagits. Exempel på beräkningar som har varit relevanta i forskningen om super- och hyperturing är bland annat beräkningar som kräver oändligt antal beräkningssteg, oändlig precision eller oändligt minne eller är icke-algoritmiska (Cotogno, 2003; Piccinini, 2012). I inledningen till kapitel 4 kommer några beräkningsmodeller och beräknare som har föreslagits vara super- eller hyperturing att presenteras.

3.5 Church-Turings hypotes

Church-Turings hypotes, här kallad CTT (från Church-Turing Thesis), uttrycker hur turingmaskiner relaterar till beräkningsbara problem. Hypotesen kan formuleras

CTT: *”Alla funktioner som är intuitivt beräkningsbara går att beräkna med någon turingmaskin.” Piccinini (2011, rapportförfattares översättning)*

Vidare förklarar Piccinini att följderna av hypotesen beror till stor del på hur den tolkas med avseende på det vaga nyckelordet *intuitivt*. Nedan följer två för okonventionell beräkning relevanta tolkningar av hypotesen.

3.5.1 Matematisk CTT

Piccinini (2011) och Copeland (2008) lyfter först och främst fram den beräkningsteoretiska tolkning, här kallad MCT (från Mathematical Church-Turing thesis), som omformulerar den intuitiva delen i termer av *effektiva rutiner*.

MCT: *”Alla funktioner som går att beräkna medelst en effektiv rutin går att beräkna med någon turingmaskin.” Piccinini (2011, rapportförfattares översättning)*

Det är denna tolkning av hypotesen som relaterar turingmaskiner, och därmed andra konventionella beräkningsmodeller, till synen på beräkningar som algoritmiska då effektiva rutiner praktiskt sett är algoritmer. Motsatsen till hypotesen, alltså att det finns algoritmer till varje turingmaskin, motiveras genom att turingmaskiners operationer är en ändlig serie av elementära operationer, det vill säga, en algoritm. Det som MCT inte täcker in är de beräkningar som inte går att beskriva som en effektiv rutin eller algoritm. Piccinini (2011)

MCT har varken bevisats eller motbevisats, men det finns argument för att dess korrekthet. Piccinini (2012) presenterar i korthet följande huvudargument för korrektheten hos hypotesen:

1. Inga motexempel i form av algoritmer som ej går att beräkna med en turingmaskin har hittats.
2. Det har visat sig att försök till att konstruera en algoritm som ej går att beräkna med en turingmaskin genom att diagonalisera över olika turingmaskiner alltid går att beräkna med någon turingmaskin.
3. Försök till att hitta andra formuleringar av MCT har lett till ekvivalenta hypoteser.
4. Turingmaskiner verkar kapabla till att utföra de beräkningssteg som en människa klarar av.

3.5.2 Fysisk CTT

PCT (Physical Church-Turing thesis) är en fysisk tolkning av CTT som beskriver hur turingmaskiner relaterar till fysiska system. Exakt vilka fysiska system som bör inkluderas i hypotesens omfång är en av variablerna att ta hänsyn till då hypotesen formuleras. Fokus kommer att hamna hos den mer restriktiva versionen, här kallad MPCT (från Modest Church-Turing thesis).

MPCT: *”Alla funktioner som går att beräkna fysiskt går att beräkna med någon turingmaskin.” Piccinini (2011, rapportförfattares översättning)*

Vad MPCT säger är att ingen fysisk beräknare kan överstiga beräkningskraften av en universell turingmaskin. Detta kräver en striktare formulering av vilka system som bör ses som beräknande. Piccinini (2012) presenterar följande (ej uttömande) lista över krav som ställs på ett system för att det ska ses som beräknande:

In- och utdata som går att läsa av utan fel. Detta exkluderar till exempel system med godtyckliga reellvärda tillstånd, såsom samlingar av himlakroppar.

Processoberoende regler som går att använda för att förutse processen utan att implementera den.

Reproducerbarhet som tillåter användaren att utföra samma beräkning flera gånger och få samma resultat. Detta exkluderar icke-deterministiska system med till exempel komponenter beroende av radioaktivt sönderfall.

Återställbarhet som tillåter användaren att utföra flera beräkningar med samma system.

Konstruerbarhet, systemet måste vara fysiskt realiserbart och möjligt att konstruera.

Pålitlighet, systemet får inte haverera mitt i en beräkning.

Piccinini (2011) påpekar även att dessa krav ställda på fysiska system är mer inkluderande än MCTs omfång, då det finns fysiska system som inte utifrån ser ut att bete sig algoritmiskt. För okonventionell beräkning betyder detta att om hypotesen är sann, så kan även icke-algoritmiska system uttryckas i en turingmaskin och därför även i en konventionell dator. Inversen till hypotesen, att det går att bygga fysiska implementationer av turingmaskiner, är inte lika slående då det konventionella datorer kan ses som praktiska turingmaskiner. Inversen är också bara sann då reservationer för minneskapacitet görs. Inom okonventionell beräkning handlar det snarare om denna invers går att uppfylla med andra fysiska system än de som används idag.

Kapitel 4

Okonventionell beräkning

Beräkningar, beräknare och beräkningsmodeller kan vara okonventionella på olika vis (Adamatzky, 2014). Ett av dessa vis uttrycks i frågan: ”Går det att tillämpa konventionella beräknare på okonventionella vis?” Såväl Piccinini (2012) som Cotogno (2003) presenterar några exempel från denna framför allt teoretiska forskning: System som utnyttjar egenskaperna hos relativistisk rymdtid för att en betraktare efter ändlig tid ska ta emot resultatet av en oändligt lång beräkning; system som utnyttjar kvantmekaniska fenomen för att skicka beräknare bakåt i tiden; och så kallade ”zenomaskiner”, hypotetiska fysiska beräknare vars varje beräkningssteg tar hälften så lång tid som det föregående. Gemensamt för såväl dessa tre specifika exempel som inriktningen i stort är strävan att genom framför allt oändligt många beräkningssteg under ändlig tid åstadkomma beräkning på realiserbar superturingnivå. Resonemanget kan skissas kort med hjälp av stopproblemet (se avsnitt 3.1.1): Efter oändligt många beräkningssteg, istället för obegränsat men ändligt många, har en beräkning antingen nått ett slut eller inte. Den praktiska konstruerbarheten av beräknare och genomförbarheten av beräkningar av detta slag ifrågasätts däremot av exempelvis Cotogno (2003) på bland annat termodynamiska grunder, och dessa okonventionella idéer hamnar således i bästa fall i kategorin hyperturing.

Ett annat sätt att närma sig okonventionella beräkningar är vilket fysiskt system som utgör en beräknare (Adamatzky, 2014). Detta perspektiv kan formuleras i frågan: ”Kan andra fysiska system än digitala datorer användas som beräknare, och till vilka beräkningar?” Exempel på detta är analoga datorer, kemiska beräknare och celler, bakterier och kvantmekaniska processer som beräknare (Penrose och Zenil, 2013; Adamatzky, 2014). Syftet med dessa beräknare behöver inte vara att uppnå turingkompletthet, men då den allmänna definitionen av beräkning, en process med start- och sluttillstånd, är så bred är det ändå fördelaktigt med kriterier för vad som krävs för att ett system ska räknas som en beräknare, så som de Piccinini (2012) föreslår (beskrivna i avsnitt 3.5.2).

Istället för att undersöka fysiska system och vilka beräkningar de är kapabla till går det att förflytta sig till modellernas abstraherade domän. Frågan som ställs då är: ”Går det att formalisera beräkningar med någon given modell, av exempelvis ett fysiskt system, och vilka i så fall?” Det intressanta med denna fråga är att en modell, i egenskap av en abstraktion, inte nödvändigtvis är bunden till en särskild implementation i ett visst fysiskt system. Modellerna kan på grund av det potentiellt realiseras i olika beräknare och öppna upp nya möjligheter eller förenkla tidigare svåra uppgifter för dem (Fernandez, 2009). Exempel på sådana modeller är neuronnät, kvantfysiska qubitar och organismers enskilda eller samlade beteenden (Siegelmann, 1999; Cavaliere och Leupold, 2010).

Av de tre frågor och forskningsområden som har redogjorts för här ovan är det framför allt de två senare som berörs och avhandlas genom exempel i detta kapitel. Beräknarna och beräkningsmodellerna som behandlas i detta kapitel är: artificiella neuronnät, i synnerhet modellen ARNN; kvantberäkning; kemisk beräkning, i synnerhet reaktion-diffusionberäkning.

4.1 Artificiella neuronnät

Artificiella neuronnät[3] är ett samlingsnamn för matematiska modeller för beräkningar inspirerade av (men inte strängt baserade på) resultat om hjärnans mekanismer producerade under 40-talet (Priddy och Keller, 2005). I sin enklaste form är de nät av processorer, som motsvarar hjärnans neuroner och beskrivs av matematiska aktiveringsfunktioner, med viktade samt riktade sammankopplingar, motsvarande synapserna. Varje ”neuron” tar emot en eller flera värden, ”signaler”, för vilka den producerar ett värde i enlighet med sin aktiveringsfunktion; denna signal skickas vidare till andra neuroner i enlighet med neuronens sammankopplingar, vars vikter förstärker eller försvagar signalen. Neuronerna i nätet är i sin tur organiserade i ”lager”, varje bestående av en eller flera neuroner: ett ”inlager” som tar emot indatan till nätet; ett ”utlager” som matar ut beräkningens resultat; och ett antal inre ”dolda lager” som står för de huvudsakliga beräkningarna. Denna uppbyggnad av flera oberoende processorer gör neuronnät kapabla till parallell beräkning (Priddy och Keller, 2005).

4.1.1 Egenskaper och tillämpningar

Ett neuronnäts beräkningskraft påverkas av tre faktorer: neuronernas aktiveringsfunktioner; förbindelsernas vikter; och nätets arkitektur (Siegelmann, 1999). Aktiveringsfunktioner såväl som vikter kan bestämmas till att vara allt från att vara binära med exempelvis värdena 1 och 0 till att tillhöra den komplexa talmängden $\mathbb{C}$ och då potentiellt kunna anta icke-reella värden; olika konfigurationer av de två bidrar till olika kraftnivåer. En arkitektonisk egenskap hos neuronnät är hur neuronerna sammantaget är förbundna till varandra: nät är antingen framkopplande (acykliska) eller återkopplande (cykliska) (Siegelmann, 1999).

En framhävande egenskap hos neuronnät är att de kan vara statiska eller dynamiska (Siegelmann, 1999). Ett statiskt nät är något så enkelt som ett nät vars förbindelsers vikter är på förhand givna konstanter. Statiska nät används med fördel till problem som kräver precisa lösningar. Värdena som det statiska nätets vikter har kan fås på olika sätt, varav ett är att utnyttja parametriska vikter hos ett dynamiskt neuronnät. Dynamiska neuronnät kan nämligen anpassa sina förbindelsers vikter och därmed lära sig att lösa problem. Denna inlärning kan ske med hjälp av exempelvis på förhand givna in-och-utvärdepar $(x, f(x))$ (så kallad övervakad inlärning) eller anpassningsregler (oövervakad inlärning) (Priddy och Keller, 2005). Vanligtvis slumpas vikterna i början av inlärning, varefter inre eller yttre mekanismer korrigerar vikternas värde med avseende på värdeparen eller anpassningsreglerna.

Förmågan till inlärning, och följaktligen mönsterigenkänning, hos neuronnät (tillsammans med deras parallellism) har varit av stort och brett intresse och modellen har därför sett tillämpningar i olika sammanhang. Kamruzzaman, Begg och Sarker (2006) ger ett flertal exempel på hur implementationer av neuronnät har använts inom finans, för bland annat konkursförutsägelser och aktiemarknadsanalyser, och inom tillverkning, för sådant som kostnadsuppskattningar, schemaläggningar och kvalitetskontroll. Ett annat exempel på tillämpningen av neuronnät är olika former av bildbehandling i bred bemärkelse, däribland förbehandling, komprimering och bildigenkänning, vilket Egmont-Petersen, de Ridder och Handels (2002) gör en genomgång av, och Jiang, Trundle och Ren (2010) behandlar delfältet medicinsk bildbehandling medelst neuronnät. Inom robotiken har neuronnät kommit till användning för programmering av robotars sensomotorik, vilket bland annat

[3]Svenska för engelskans ”Neural Network” (http://www.datatermgruppen.se/ordlista.html)

Torras (2005) redovisar. Vad gäller tillämpningsform kan neuronnät i allmänhet simuleras på, eller konstrueras som konventionella digitala datorer[4] såväl som särskilt designad hårdvara, såsom ”neurodatorer” (Kussul, Baidyk och Wunsch, 2010).

4.1.2 Artificial Recurrent Neural Network

Med avseende på frågan om superturingberäknare är en kandidat modellvarianten av neuronnät som på engelska kallas Artificial Recurrent Neural Network (ARNN, ”artificiellt återkopplande neuronnät”) (Siegelmann, 1999). Denna kan nämligen konfigureras till både att vara ekvivalent med turingmaskinen såväl som hyperturing. Konfigurationen som krävs är att nätet är återkopplande, att dess neuroners aktiveringsfunktioner är kontinuerliga och att sammankopplingarna är enkelriktade och har rationella ($\mathbb{Q}$) respektive (icke-rationella) reella ($\mathbb{R} \setminus \mathbb{Q}$) vikter. Turingmaskiner är som bekant begränsade i vilka problem, kodade som binära strängar, de kan beräkna, medan ARNN med reella vikter principiellt kan beräkna alla problem (Siegelmann, 1999; Cotogno, 2003). Om kraften hos ARNN menar Siegelmann dessutom att ”No possible abstract analog device can have more computational capabilities than ARNN” (Siegelmann, 1999, s. xi).

Efter detta återstår med avseende på MPCT i huvudsak frågan om ARNN, med sin fulla hyperturingkraft, är fysiskt realiserbar och praktiskt användbar. Så menar Siegelmann (1999) att fallet är med ARNN, och motiverar det med förmågan att koda nätets in- och utvärden som binära strängar, och därmed begränsa icke-rationella värden till nätets ”insida”, och att vikternas reella värden, som potentiellt kräver oändlig precision, därtill går att koda med ändlig precision. Detta bemöter dock Piccinini (2012) med att reella värden oavsett kodning behöver oändlig precision i något led, och för detta krävs redan superturingkraft. Även Cotogno stämmer in i detta och tillägger att ARNN, likt andra beräknare som hanterar analoga värden, förlorar mycket om inte all sin beräkningskraft, och drar därför slutsatsen att ”the true concern with analog computers should be to ensure that implementations are reliable enough to maintain, rather than to surpass, the power of [Turing Machines]” (Cotogno, 2003, s. 204).

4.2 Kvantberäkning

Kvantberäkning är en kombination av informationsteori och kvantmekanik (Rieffel och Polak, 2011). Idén om kvantdatorer kom till när Feynman (1982) med flera märkte att ett kvantmekaniskt fenomen kallat sammanflätning inte går att simulera på ett effektivt sätt i en vanlig dator. En kvantdator var alltså från början tänkt som en kvant*simulator*, där istället för att räkna med en vanlig dator kunde man använda ett fysiskt system av partiklar för att simulera ett annat (Ivancevic, 2008). Tanken om att utnyttja kvantmekaniska fenomen i fysiska system, även för att utföra andra sorters beräkningar, för att överträffa turingmaskiners beräkningskapacitet hade fötts. Deutsch och Jozsa (1992) skapade en av de första kvantalgoritmer som kunde bevisas ha lägre tidskomplexitet (storleksordning av tidsåtgången för genomförandet av en specifik beräkning) än någon möjlig konventionell algoritm. Senare kom Shor (1994) med sin faktoriseringsalgoritm, som kraftigt ökade intresset för kvantberäkning, eftersom primtalsfaktorisering har betydande praktiska tillämpningar. Bland annat bygger den mycket välanvända krypteringsalgoritmen RSA sin säkerhet på att primtalsfaktorisering av tillräckligt stora tal tar för lång tid för att vara praktiskt genomförbart (Katzenbeisser, 2001).

[4] Se `https://grey.colorado.edu/emergent/index.php/Comparison_of_Neural_Network_Simulators` [Hämtad 12:e maj 2015] för exempel på mjukvaruimplementationer av neuronnät.

4.2.1 Qubitar

Kvantberäkning skiljer sig från konventionell beräkning genom att beräkningarna sker med hjälp av qubitar[5], istället för bitar. En bit kan endast anta värdena 1 och 0, men en qubit kan anta en superposition av dessa. Precis som att en bit kan representeras av olika saker (spänningsnivåer, hål i hålkort, ettor och nollor skrivna på ett papper etc.), kan qubitar realiseras av olika saker (bland annat spin av en atomkärna (Jelezko, Gaebel, Popa, Domhan, Gruber och Wrachtrup, 2004), polarisationsriktningen av en foton (Rieffel och Polak, 2011) eller en supraledande krets (Kerman, 2010)).

Den bland allmänheten kända beskrivningen att en qubit kan vara ”både ett och noll samtidigt” är något missvisande. Som exempel väljs en fotons polarisation som får representera en qubit; om vertikal polarisation motsvarar en nolla, och horisontell polarisation motsvarar en etta, innebär polarisation i en vinkel att fotonen är i en superposition av att vara etta och nolla. Ett kvantmekaniskt fenomen gör att om ett polarisationsfilter läggs i vertikalriktning framför fotonen, så kommer fotonen att ta sig igenom om den är polariserad i vertikalriktning. Motsvarande, om fotonen är polariserad i horisontalriktningen, vinkelrät mot filtret, kommer den att fastna. Ju närmare vinklarna är varandra, desto högre är sannolikheten att fotonen tar sig igenom filtret, den kommer aldrig att att komma igenom bara delvis. Tar fotonen sig igenom, så kollapsar[6] den i vertikalriktningen (detta motsvarar alltså mätning med resultatet noll). Informationen om fotonens ursprungsläge går förlorad, därmed går det endast att få ut motsvarande en bit information från en qubit vid mätning. Denna information är icke-deterministisk, och många kvantalgoritmer ger korrekt resultat endast med en viss sannolikhet (Rieffel och Polak, 2011).

Först när flera qubitar samverkar i en kvantdator kommer dess fördelar fram. För att specificera tillståndet för n bitar, behövs endast n ettor eller nollor. Qubitar däremot kan vara sammanflätade, ett fenomen som saknar motsvarighet i den makroskopiska världen, vilket gör att det inte räcker med att beskriva varje qubits tillstånd var för sig. n qubitar kan vara i en superposition av upp till 2^n tillstånd samtidigt, det krävs alltså 2^n värden för att specifiera n stycken qubitars tillstånd. Vid mätning kollapsar det sammanflätade tillståndet och endast n bitar information kan fås ut, men mätning görs först när beräkning är utförd (Rieffel och Polak, 2011).

4.2.2 Beräkningskraft och beräkningskapacitet

Kvantdatorer har fördelen gentemot konventionella beräknare att de kan generera äkta slumpmässiga tal, men pseudo-slumpmässiga strängar av tal kan göras godtyckligt nära äkta slumpmässiga dito (Cotogno, 2003). Därmed ökar inte förmågan att generera slumpmässiga tal en dators beräkningskraft, och kvantdatorer har alltså lika stor beräkningskraft som turingmaskiner (Lloyd, 1993).

Det finns två vanliga modeller för kvantberäkning, varav en är kvantturingmaskiner, som är kvantdatorers (betydligt mer komplicerade) motsvarigheter till turingmaskiner (Bernstein och Vazirani, 1997). Den andra är kvantkretsar[7], som är motsvarigheten till att konventionella datorer består av en logisk krets som går att uppdela i en serie av logiska portar, som beskrivet i 1.2. På samma sätt kan kvantalgoritmer delas i en serie operationer som avbildar kvanttillstånd på kvanttillstånd (Ivancevic, 2008). För varje funktion som

[5]Från engelskans ”quantum bit”

[6]vid mätning av ett superponerat kvanttillstånd ändras kvanttillståndet till det uppmätta värdet och resterande delar av det superponerade tillståndet försvinner. Fenomenet kallas för kollaps. Det går alltså inte att mäta ett kvanttillstånd utan att påverka den (Rieffel och Polak, 2011).

[7]Från engelskans "quantum circuit".

går att beräkna under polynomiell tid (där tidsåtgången begränsas av ett polynom med storleken av indatan som variabel, till skillnad från till exempel en exponentialfunktion) på en kvantturingmaskin, finns en kvantkrets av polynomiell storlek, och det finns en universiell kvantturinmaskin som kan simulera en godtycklig annan kvantturingmaskin med endast en polynomiellt ökad tidsåtgång (Yao, 1993).

Exakt vilka problem som kan lösas effektivare med kvantalgoritmer jämfört med konventionella dito är i dagsläget inte känt (Nielsen och Chuang, 2011). Shors algoritm för kvantdatorer kan utföra primtalsfaktorisering inom polynomiell tid. Det existerar inga kända algoritmer för turingmaskiner som kan göra detsamma, men det är inte bevisat att det inte är möjligt. Men, som Rieffel och Polak (2011, s. 3) uttrycker saken: "even in the unlikely event that a polynomial-time classical algorithm is found for this problem, it would be an indication of the elegance and effectiveness of the quantum information theory point of view that a quantum algorithm, in spite of all the unintuitive aspects of quantum mechanics, was easier to find".

4.2.3 Problem med kvantberäkning och kvantberäkning idag.

Ett av de största problemen med implementation av kvantdatorer är dekoherens. Enkelt uttryckt handlar det om växelverkan mellan qubitar och omgivningen, vilket sker om qubitar inte är perfekt isolerade (DiVincenzo, 1995). För att minimera störningar behövs även väldigt låga temperaturer, i storleksordningen 1 K (Barnes, 1998).

Ett fåtal demonstrationer av småskaliga kvantdatorer har gjorts, bland andra faktoriserade Politi, Matthews och O'Brien (2009) siffran 15 till 3 och 5 med hjälp av 4 qubitar och Shors algoritm. År 2011 sålde D-Wave Systems sin första kvantdator, med 128 qubitar, till företaget Lockheed Martin. År 2013 köpte Quantum Artificial Intelligence Laboratory, bestående av NASA, Google och USRA, dess efterföljare D-Wave Two med 512 qubitar (NASA, 2015). Det är dock oklart om dessa verkligen har någon fördel jämfört med konventionella datorer, och D-wave Systems har fått stark kritik. Bland annat Shin, Smith, Smolin och Vazirani (2014) presenterar en klassisk modell som de påstår ha samma prestanda.

4.3 Reaktions-Diffusionsberäknare

Reaktions-Diffusionsberäknare (RD-beräknare) är en underkategori till kemiska beräknare som utför beräkningar genom att en eller flera kemiska substanser som diffunderar och/eller reagerar med sig själv och/eller varandra. Information i en RD-beräknare utgörs av koncentration och tillstånd av den kemiska lösningen som används vid varje position i beräknaren. Beräkningsförloppet är de spontana reaktioner som sker den kemiska lösningen (Adamatzky och Costello, 2012). Till exempel finns RD-beräknare som använder en kemisk lösning som oscillerar mellan två tillstånd, en så kallad Belousov-Zhabotinsky-reaktion (BZ-reaktion). Olika egenskaper hos BZ-lösningen kan bidra med ytterligare funktionalitet hos beräknaren. En BZ-lösning med visuellt urskiljbara tillstånd går att läsa av med ögat eller en kamera. Om lösningen är ljuskänslig så är det möjligt att genom belysning manipulera beräkningen under tiden som den sker.

4.3.1 Beräkningskraft

Beräkningskraften hos denna sortens överstiger inte den hos turingmaskiner eftersom beräknare på denna formen uppfyller den fysiska tolkning av Church-Turings hypotes som

presenterades i avsnitt 3.5. Ett resultat som motiveras utav att RD-beräknaren praktiskt uppfyller alla kraven som ställs.

4.3.2 Arkitektur- och kollisionsbaserade logiska portar

En av infallsvinklarna till att konstruera en universell RD-beräknare är att likt konventionella beräknare konstruera en samling av logiska portar. Adamatzky och Costello (2012) klassificerar följande två tillvägagångssätt för att uppnå logiska portar i ett BZ-medium.

Arkitekturbaserad beräknare har en förutbestämd statisk struktur med föränderliga inre tillstånd. Ett konventionellt exempel på detta är hårddiskar som har en statisk form med varierande magnetiska laddningar. En arkitekturbaserad RD-beräknare kan utgöras av geometriska begränsningar i rummet BZ-vågorna propagerar genom. Adamatzky, De Lacy Costello och Asai (2005, kap.3) visar att det går att välja dessa begränsningar som resulterar i logiska portar.

Kollisionsbaserade beräknare har i kontrast till arkitekturbaserade beräknare inga förutbestämda positioner vid vilka beräkning sker. Beräkning sker spontant på icke förutbestämd tid och plats när två molekyler reagerar med varandra. En analog parallell är biljardbollar som färdas över ett bord, dessa ändrar sin befintliga bana först då och där de stöter in i en annan boll. Även för kollisionsbaserade RD-beräknare är det möjligt att konstruera logiska portar och därmed teoretiskt möjligt att uppnå universell beräkningskraft (Adamatzky, 2004).

4.3.3 Problemspecifika beräknare

Istället för att konstruera en universell beräknare genom kluster av logiska portar så finns det RD-beräknare vars mål är att utföra en specifik beräkning som löser ett problem eller en kategori av problem. Beräkningar som tar lång tid för konventionella beräknare att utföra och där RD-beräknares parallellism kan komma till nytta.

En sådan kategori av beräkningar är bildbehandling Kuhnert, Krinsky och Agladze (1989) visade hur bilder i gråskala kan överföras till en ljuskänslig BZ-lösning och därefter manipuleras till viss grad.

En annan tillämpning av RD-beräkning är de så kallade mikrokemimekaniska system som Voigt, Greiner, Allerdissen, Richter, Henker och Völp (2014) föreslår. Systemen tar formen av chip uppbyggda av kemiska komponenter. Sensorer som reagerar på biologiska signalämnen samverkar med kemiska transistorer för att omvandla signalen till den önskade utsignalen. Jämfört med ett konventionell chip så tjänar det mikrokemiska systemet på att inte behöva översätta från biologiska till elektriska signaler.

4.3.4 Nackdelar

RD-beräknare kan på grund av BZ-lösningars begränsade propagationshastighet inte komma upp i samma storleksordning av elementära operationer som en konventionell beräknare. Teoretisk sett skulle BZ-lösningar med snabbare propagationshastigheter kunna råda bot på detta (Adamatzky et al., 2005, kap. 1). En annan möjlighet som Asai (2012) utforskat är att bygga ett RD-chip av semikonduktorer i en slags fysiskt neuronnät som härmar det kemiska RD-beteendet. Det är möjligt att simulera ett sådant nätverk i en konventionell dator men som Asai påpekar sker inmatning seriellt i konventionella processorer medans det är möjligt att nätprocessorn kan ta emot parallell inmatning.

Kapitel 5

Diskussion

I vitboken ”More-than-Moore” (Arden, Brillouët, Cogez, Graef, Huizing och Mahnkopf, 2010) påpekas det att ett beräknande systems användbarhet inte alltid är direkt kopplad till systemets beräkningsprestanda. Genom att fokusera på användargränssnitt, gränssnitt mot andra system och andra egenskaper som inte är relaterade till beräkningsprestanda kan dock systemet ändå förbättras. I sådana tillämpningar är det möjligt att okonventionell beräkning spelar en roll. Parallellt med ett sådant resonemang går det att fråga sig om den utvecklingen Moores lag beskriver kan fortsätta obehindrat. En stagnation av utveckling av transistorbaserade beräknare skulle ytterligare motivera forskning och utveckling av okonventionell beräkning.

En annan möjlighet är att okonventionella beräknare, på samma sätt som konventionell beräkning, implementeras på ställen där automatiserad beräkning inte tidigare använts och därmed motiverar sin användbarhet. Exempel på detta skulle exempelvis vara, som nämnt i förra stycket, olika gränssnitt mot andra system, såsom biologiska vävnader. På liknande sätt kan okonventionella modeller, precis som olika konventionella beräkningsmodeller, som exempelvis turingmaskinen och λ-kalkylen, har inspirerat olika programmerings- och problemlösningsparadigm, bidra till eller utlösa utvecklingen av nya metoder och verktyg för den konventionella beräkningen.

Nedan följer en diskussion kring de tre beräknare som presenterades i föregående kapitel.

5.1 Artificiella neuronnät

Neuronnät är den av de tre presenterade okonventionella beräknare och beräkningsmodeller som står sig stadigast jämte moderna konventionella dito, med nästan lika många decennier på ryggen. De exempel som presenterades i slutet av avsnitt 4.1.1 är inte uttömande och nya tillämpningar inom såväl gamla som nya områden utvecklas och upptäcks kontinuerligt. Förvisso tävlar hårdvarurealiseringar av neuronnät i dagsläget inte med konventionella datorer i universalitet, trots möjligheten att såväl modellera som realisera turingkompletta neuronnät. En av de största faktorerna i det är neuronnätens relativt svåra programmerbarhet i kontrast mot digitala datorer: digitala datorers uppdelning i processor och arbetsminne, läshuvud och band (se 3.1.1), såväl som deras binära kodning, för att inte tala om den uppsjö av operativsystem och programmeringsspråk som finns, gör datorn relativt enkel att programmera. Neuronnät saknar såväl uppdelningen i processor och minne från varandra som förmildrande omständigheter såsom enkel programmering och hjälpmedel för det.

Att det förhåller sig på det viset har däremot som visat inte hindrat neuronnät från att tillämpas i en mängd områden. Detta visar dels på deras användbarhet som sådan men dels även exemplifierar en poäng som lyftes i avsnitt 3.3.1: en beräknares eller beräkningsmodells beräkningskraft säger lite om något om dess faktiska och potentiella användningsområden. Neuronnät har som nämnt just visat sig vara användbara för uppgifter som historiskt har varit svåra för digitala datorer, nämligen inlärning och igenkänning. En

grundbrytande konsekvens av detta tåls att jämföra med rollen som turingmaskiner och digitala datorer har spelat för den vetenskaplig och samhälleliga utvecklingen. Om den konventionella beräkningsteorin och tekniken formaliserade och automatiserade utförandet av beräkningar (se 3.1.1) så kan den okonventionella beräkningen, via neuronnät och deras förmåga till inlärning och igenkänning, automatisera utformandet av själva beräkningarna som ska utföras.

5.2 Kvantdatorer

En lyckad implementering av en kvantdator med tillräckligt många qubitar skulle ha långtgående konsekvenser för samhället. Inte minst eftersom det skulle ge innehavarna av kvantdatorn möjlighet att med hjälp av Shors algoritm avläsa datatrafik som var krypterad med RSA. RSA skulle alltså behöva ersättas med en annan algoritm, eller så skulle avsevärt lägre säkerhetsnivå på internet behöva accepteras.

Kvantdatorer har visat sig ha vissa specifika algoritmer med mycket lägre tidskomplexitet jämfört med motsvarande algoritmer för konventionella datorer. Detta väcker frågan om kvantdatorer skulle kunna underlätta även på andra beräkningstunga områden. Det finns fortfarande stora problem vid implementering, men det som talar för kvantdatorer är att stora, ekonomiskt starka aktörer driver forskning på området.

Även om människan aldrig skulle lyckas bygga kvantdatorer stora nog för att ha en tidsmässig fördel gentemot konventionella datorer, är de intressanta ur ett beräkningsteoretiskt perspektiv. De är inte enbart hypotetiska beräknare, så som zenomaskiner som nämndes i 4, utan de är bevisbart möjliga, och de har polynomiella algoritmer för problem som bevisbart inte går att lösa med turingmaskiner under polynomiell tid.

I vardagligt användande kommer kvantdatorer med största sannolikhet inte att ersätta konventionella datorer. Om kvantdatorer överhuvudtaget kommer byggas, kommer de med största sannolikhet alltid att vara större och känsligare än vad som är rimligt att försöka bygga in i en bärbar enhet. Även nyttan med kvantberäkning är väldigt begränsad i vardaglig användning, då det är endast vid lösning av väldigt specifika problem som kvantberäkning är att föredra. I vardaglig användning är responsivitet, pålitlighet, låg strömförbrukning och pris viktigare faktorer.

5.3 Reaktion-Diffusionberäknare

Trots den parallellism som råder vid kemiska reaktioner har de RD-beräknare som undersökts i denna studie inte varit i närheten av konventionella beräkningshastigheter, till viss del beroende på materialbegränsningar. Utveckling av snabbare BZ-lösningar eller artificiella RD-beräknare skulle teoretiskt sett kunna råda bot på detta. Men det som gör RD-beräkning intressant är möjligheten att parallellt läsa och skriva data på ett sätt som inte går att implementera i konventionella beräknare.

En annan intressant egenskap hos kemiska beräknare i allmänhet är närheten till andra områden så som biologi. I de mikrokemimekaniska systemen samaverkar kemiska transistorer med biologiska komponenter för att skapa ett system som är beräkningsmässigt mindre resurskrävande än en motsvarande digitala beräknare eftersom den biologiska indatan inte behöver översättas till elektriska impulser.

Eftersom RD-beräknare i alla utformningar som undersökts i denna studie inte kunnat mäta sig med mångsidigheten och beräkningshastigheten hos konventionella datorer så är det mest troligt att, om RD-beräknare implementeras, så är det som en problemspecifik

beräknare anpassad för sin uppgift. Möjligtvis som uppgiftsdedikerad komponent i en annars konventionell dator.

5.4 Avslutning

Avslutningsvis, för att poängtera värdet av att utforska beräkning, konventionell som okonventionell, och dess möjligheter går det att citera matematikerna Ada Lovelace kommentarer från sitt arbete med en tidig analog dator:
”The operating mechanism can even be thrown into action independently of any object to operate upon (although of course no result could then be developed). Again, it might act upon other things besides number, were objects found whose mutual fundamental relations could be expressed by those of the abstract science of operations, and which should be also susceptible of adaptations to the action of the operating notation and mechanism of the engine. Supposing, for instance, that the fundamental relations of pitched sounds in the science of harmony and of musical composition were susceptible of such expression and adaptations, the engine might compose elaborate and scientific pieces of music of any degree of complexity or extent” (King och Lovelace, 1843).

Litteraturförteckning

Adamatzky, A. (2004). Collision-based computing in Belousov-Zhabotinsky medium. *Chaos, Solitons and Fractals*, *21*(5), 1259–1264.

Adamatzky, A. (2014). Unconventional computing. *International Journal of General Systems*, *43*(7), 671–672.

Adamatzky, A. och Costello, B. D. L. (2012). *Reaction Diffusion Computing*, (pp. 1897–1920). Berlin, Heidelberg: Springer Berlin Heidelberg.

Adamatzky, A., De Lacy Costello, B., och Asai, T. (2005). *Reaction-diffusion computers*. Amsterdam; Boston: Elsevier.

Arden, W., Brillouët, M., Cogez, P., Graef, M., Huizing, B., och Mahnkopf, R. (2010). More-than-moore. Hämtad från `http://www.itrs.net/ITRS%201999-2014%20Mtgs,%20Presentations%20&%20Links/2010ITRS/IRC-ITRS-MtM-v2%203.pdf`.

Asai, T. (2012). *Unconventional Computing, Novel Hardware for*, (pp. 3260–3279). New York, NY: Springer New York.

Barnes, S. E. (1998). Efficient quantum computing on low temperature spin ensembles. *arXiv preprint quant-ph/9804065*.

Bernstein, E. och Vazirani, U. (1997). Quantum complexity theory. *SIAM J. Comput.*, *26*(5), 1411–1473.

Cavaliere, M. och Leupold, P. (2010). Computing by observing changes. In F. Peper, H. Umeo, N. Matsui, och T. Isokawa (Eds.), *Natural Computing*, volume 2 of *Proceedings in Information and Communications Technology* (pp. 133–140). Tokyo: Springer Japan.

Chalmers, D. J. (1996). Does a rock implement every finite-state automaton? *Synthese*, *108*(3), 309–333. Synthese.

Copeland, B. J. (2008). The church-turing thesis. In E. N. Zalta (Ed.), *The Stanford Encyclopedia of Philosophy* (Fall 2008 ed.). Hämtad från `http://plato.stanford.edu/archives/fall2008/entries/church-turing/`.

Copeland, J. B. och Proudfoot, D. (1999). Alan turing's forgotten ideas in computer science. *Scientific American*, *280*(4), 99–103.

Cotogno, P. (2003). Hypercomputation and the physical church-turing thesis. *The British Journal for the Philosophy of Science*, *54*(2), 181–223.

Davis, M. (2006). *The Church-Turing Thesis: Consensus and opposition*, volume 3988, (pp. 125–132). Berlin, Heidelberg: Springer Berlin Heidelberg.

Deutsch, D. och Jozsa, R. (1992). Rapid solution of problems by quantum computation. *Proceedings: Mathematical and Physical Sciences*, *439*(1907), pp. 553–558.

DiVincenzo, D. P. (1995). Quantum computation. *Science*, *270*(5234), pp. 255–261.

Egmont-Petersen, M., de Ridder, D., och Handels, H. (2002). Image processing with neural networks—a review. *Pattern recognition*, *35*(10), 2279–2301.

Fernandez, M. (2009). *Models of computation: an introduction to computability theory*. Dordrecht: Springer.

Feynman, R. (1982). Simulating physics with computers. *International Journal of Theoretical Physics*, *21*(6-7), 467–488.

Harel, D. (2000). *Computers Ltd: What They Really Can't Do*. Oxford: Oxford University Press.

Harel, D. och Feldman, Y. A. (2004). *Algorithmics: the spirit of computing*. New York; Harlow, England: Addison Wesley.

Hodges, M. S. och (e-book collection), B. I. . B. (2007). *Computers: systems, terms, and acronyms, 17th edition*. Casselberry, Fla: SemCo Enterprises.

Hopcroft, J. E., Motwani, R., och Ullman, J. D. (2006). *Introduction to automata theory, languages, and computation*. Harlow: Pearson Addison-Wesley.

Ivancevic, V. G. (2008). *Quantum leap: From Dirac and Feynman, across the universe, to human body and mind*. Hackensack, NJ: World Scientific.

Jelezko, F., Gaebel, T., Popa, I., Domhan, M., Gruber, A., och Wrachtrup, J. (2004). Observation of coherent oscillation of a single nuclear spin and realization of a two-qubit conditional quantum gate. *Phys. Rev. Lett.*, *93*, 130501.

Jiang, J., Trundle, P., och Ren, J. (2010). Medical image analysis with artificial neural networks. *Computerized Medical Imaging and Graphics*, *34*(8), 617–631.

Kamruzzaman, J., Begg, R., och Sarker, R. A. (2006). *Artificial neural networks in finance and manufacturing*. Hershey, Pa: Idea Group Pub.

Katzenbeisser, S. (2001). *Recent Advances in RSA Cryptography*, volume 3. Boston, MA: Springer US.

Kerman, A. J. (2010). Metastable superconducting qubit. *Phys. Rev. Lett.*, *104*, 027002.

King, B. och Lovelace, A. (1843). Notes by the translator of the sketch of the analytical engine invented by charles babbage, by lf menabrea. *Scientific Memoirs*, *3*, 666–731.

Koffman, E. B. och Wolfgang, P. A. T. (2010). *Data structures: abstraction and design using Java*. Hoboken, NJ: John Wiley.

Kozen, D. C. (1997). *Automata and Computability*. New York, NY: Springer New York.

Kuhnert, L., Krinsky, V. I., och Agladze, K. I. (1989). Image processing using light-sensitive chemical waves. *Nature*, *337*(6204), 244–247.

Kussul, E. M., Baidyk, T., och Wunsch, D. C. (2010). *Neural networks and micromechanics*. Heidelberg: Springer.

Lloyd, S. (1993). Quantum-mechanical computers and uncomputability. *Phys. Rev. Lett.*, *71*, 943–946.

Mack, C. A. (2006). *Moore's Law*, (pp. 86–86). SPIE Press.

Mitchell, J. C. (2003). *Concepts in programming languages*. Cambridge: Cambridge Univ. Press.

Moore, C. och Mertens, S. (2011). *The nature of computation*. Oxford: Oxford University Press.

NASA (2015). Quantum artificial intelligence laboratory. Hämtad från `http://www.nas.nasa.gov/quantum/quantumcomp.html`.

Nielsen, M. A. och Chuang, I. L. (2011). *Quantum Computation and Quantum Information: 10th Anniversary Edition* (10th ed.). New York, NY, USA: Cambridge University Press.

Penrose, R. och Zenil, H. (2013). *A computable universe: Understanding and exploring nature as computation*. New Jersey: World Scientific.

Piccinini, G. (2011). The physical church–turing thesis: Modest or bold? *The British Journal for the Philosophy of Science*, *62*(4), 733–769.

Piccinini, G. (2012). Computation in physical systems. In E. N. Zalta (Ed.), *The Stanford Encyclopedia of Philosophy* (Fall 2012 ed.). Hämtad från `http://plato.stanford.edu/archives/fall2012/entries/computation-physicalsystems/`.

Politi, A., Matthews, J. C. F., och O'Brien, J. L. (2009). Shor's quantum factoring algorithm on a photonic chip. *Science*, *325*(5945), p. 1221.

Priddy, K. L. och Keller, P. E. (2005). *Artificial neural networks: an introduction*, volume 68. Bellingham, Wash: SPIE.

Rieffel, E. och Polak, W. (2011). *Quantum computing: A gentle introduction*. Cambridge, Mass: MIT Press.

Shin, S. W., Smith, G., Smolin, J. A., och Vazirani, U. (2014). How "quantum"is the d-wave machine?

Shor, P. W. (1994). Algorithms for quantum computation: discrete logarithms and factoring. (pp. 124–134).

Siegelmann, H. (1999). *Neural networks and analog computation: Beyond the Turing limit*. Boston, MA: Birkhäuser Boston.

Siegelmann, H. T. (1995). Computation beyond the Turing limit. *Science*, *268*(5210), 545–548.

Sipser, M. (2012). *Introduction to the Theory of Computation*. Cengage Learning.

Torras, C. (2005). Natural inspiration for artificial adaptivity: some neurocomputing experiences in robotics. In *Unconventional Computation* (pp. 32–45). Springer.

Turing, A. M. (1937). On computable numbers, with an application to the entscheidungsproblem. *Proceedings of the London Mathematical Society*, *s2-42*(1), 230–265.

Turing, A. M. (1939). Systems of logic based on ordinals. *Proceedings of the London Mathematical Society*, *2*(1), 161–228.

Voigt, A., Greiner, R., Allerdissen, M., Richter, A., Henker, S., och Völp, M. (2014). Towards computation with microchemomechanical systems. *International Journal of Foundations of Computer Science*, *25*(4), 507–523.

Yao, A. C.-C. (1993). Quantum circuit complexity. In *Foundations of Computer Science, 1993. Proceedings., 34th Annual Symposium on*, (pp. 352–361).

Bilaga A

Arbetsfördelning

Arbetet i det projekt som denna rapport summerar har till stor del varit gemensamt över projektgruppen. Till exempel har ansvaret att skriva inlägg i projektdagboken roterat mellan gruppens medlemmar. Ansvarsområden som varit tilldelade en specifik person är följande: Viktor har varit LaTeX-ansvarig och bland annat stått för problemlösning och teknisk utformning av rapport-dokumentet. Elias har designat de figurer som finns i rapporten. László har skött bokning av grupprum samt kommunikation med handledare, examinator och fackspråk.

Nedan följer en lista över huvudsaklig författare för varje avsnitt i rapporten. Om ej annat anges så är texten baserad på gemensamma litteraturstudier och diskussioner. Undantagen är avsnitt 4.0-4.3 samt 5.1-5.3 där respektive författarna själva stått för efterforskningarna.

Sammandrag László

Abstract László

Förord Viktor

1. Inledning

1.0-1.3 Viktor baserat på tidigare material från László.
1.4-1.5 Alla
1.6 Viktor

2. Metod

2.0 Viktor

3. Konventionella beräknare

3.0 László
3.1-3.3 László baserat tidigare på material från Elias
3.4 László
3.5 Viktor

4. Okonventionella beräknare

4.0 László
4.1 László
4.2 Elias
4.3 Viktor

5. Diskussion

5.0 Viktor
5.1 László
5.2 Elias
5.3 Viktor
5.4 László