Poglavje 3

Reverzibilno procesiranje

Procesiranje (angl. *computation*) v splošnem smatramo za abstraktno interpretacijo dinamičnega fizičnega procesa v kontekstu njegovih stanj in prehajanja med njimi [21]. Osnove za izvedbo procesiranja so sledeče:

- obstajati mora robustna preslikava med modelom procesiranja in fizičnim sistemom (med logičnimi vrednostmi in vrednostmi nosilcev stanj), ki mora biti stabilna in neobčutljiva na različne motnje ali perturbacije;
- fizični sistem in njegov model moramo pred delovanjem - dinamiko spreminjanja stanj (angl. *evolve*) inicializirati v njuni začetni stanji;
- po inicializaciji začetnih stanj sistem ali model začneta spreminjati svoja stanja in tvorita zaporedje stanj, katerega elementi so pripadniki množice možnih stanj; pri tem sistem in model lahko dosežeta neko končno stanje, katerega s pomočjo vnaprej definirane dinamike ni več mogoče zapustiti;

Dinamiko sistema z *diskretnim prostorom* procesnih stanj lahko predstavimo z usmerjenimi grafi, kjer so stanja sistema predstavljena z *vozlišči*, prehajanja med stanji pa z *usmerjenimi povezavami*.

3.1 Reverzibilnost dinamičnega procesa

Za *reverzibilen proces* (angl. *time invertible* ali *reversible process*) smatramo vsak proces, ki omogoča *obrnljivost dinamike* opazovanega procesa. Če dinamiko opišemo z eksaktno definiranimi stanji, bi slednje pomenilo, da reverzibilen proces omogoča prehajanje na osnovi vhoda in obstoječega stanja v novo stanje, istočasno pa tudi prehajanje iz novega stanja v predhodno stanje.

Če je dinamičen proces v časovnem smislu diskreten, prehajanje v novo stanje $q(t+1)$ zapišemo z izrazom

$$q(t+1) = f(I(t), q(t)), \tag{3.1}$$

če pa je proces v časovnem smislu zvezen, pa prehajanje v novo stanje $q(t_{i+1})$ zapišemo z izrazom

$$q(t_{i+1}) = f(I(t_i), q(t_i)), \tag{3.2}$$

kjer sta $q(t)$ in $q(t_i)$ stanji procesa v času t ali t_i, $I(t)$ in $I(t_i)$ vhodna dogodka v času t ali t_i, f pa preslikovalna funkcija, ki vhodni dogodek in obstoječe stanje preslika v novo stanje ($q(t+1)$ ali $q(t_{i+1})$) v času $t+1$ ali t_{i+1}. Oba izraza veljata tako za klasične ireverzibilne, kot tudi za reverzibilne dinamične procese. Pri tem za reverzibilne procese veljata še dodatna izraza

$$q(t) = f^{-1}(q(t+1)), \tag{3.3}$$

$$q(t_i) = f^{-1}(q(t_{i+1})), \tag{3.4}$$

kar pomeni, da informacija o predhodnem stanju ni več "izgubljena", temveč "konzervirana" v času.

V viru [22] najdemo vzorčni idealizirani primer s področja hipotetične idealne fizične reverzibilnosti. Če bi bile kemijske reakcije izgorevanja pogonskega goriva vozila reverzibilne, bi tako obstajala možnost, da bi vozilo v eni smeri vožnje gorivo trošilo, v obratni smeri vožnje pa sintetiziralo - proizvajalo na osnovi reverzibilnosti kemijskih reakcij izgorevanja. Tako bi se po povratku z izleta v rezervoarju vozila venomer nahajalo toliko goriva, kot pri odhodu.

3.2 Vrste reverzibilnosti

Ločujemo med *fizično* in *logično* reverzibilnostjo. Dinamičen proces je fizično reverzibilen, če njegova dinamika skozi čas rezultira v nezvečanje fizične entropije. V praksi fizično reverzibilnost kot lastnost poseduje proces, pri katerem ne prihaja do disipacije energije v toplotni obliki [23]. V realnem svetu takšnih procesov (še) ne poznamo, lahko pa zvečanje fizične entropije, ki jo disipacija prinaša, poljubno minimiziramo. Po Rolfu Landauerju, pionirju teorije reverzibilnosti, je osnovni pogoj za fizično reverzibilen proces njegova logična reverzibilnost. Deterministične decizijski[1] proces je logično reverzibilen, če je njegova prevajalna funkcija *bijektivna*[2].

3.3 Reverzibilnost logičnih funkcij

Reverzibilnost procesiranja je metodološki pristop, ki posega na nivo enostavnih logičnih funkcij. Slednje naj bi bile v računalnikih prihodnosti *reverzibilne*. Za začetek si oglejmo definicijo reverzibilne logične funkcije, povzeto po viru [24].

[1] Pojem decizije enačimo s pojmom odločitve (http://dis-slovarcek.ijs.si/).

[2] Prevajalna funkcija ali preslikava $f : A \to B$ je bijektivna natanko tedaj, ko je injektivna in surjektivna hkrati. Vsak element iz množice B je slika natanko enega elementa iz množice A.

x	$f_{ID}(x)$
0	0
1	1

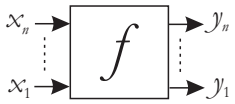
x	$f_{NEG}(x)$
0	1
1	0

Tabela 3.1: Pravilnostni tabeli enovhodnih funkcij identitete (levo) in negacije (desno). Obe funkciji sta tipa (1,1) in reverzibilni.

Definicija 1 *Logična funkcija $f(x_1, x_2, ..., x_n)$ z n Booleanovimi vhodnimi spremenljivkami je reverzibilna, če sta izpolnjena naslednja pogoja:*

- *število izhodnih spremenljivk k opazovane logične funkcije f je enako številu njenih vhodnih spremenljivk n ($n = k$);*
- *vsak vektor vrednosti vhodnih spremenljivk se preslika v njemu lasten vektor izhodnih vrednosti, ki je tako edinstven in drugačen od ostalih;*

Iz povedanega sledi, da mora biti funkcija f bijektivna funkcija (angl. *one to one mapping*), ki ima enako število vhodov in izhodov. V nadaljevanju se bomo na logične funkcije sklicevali z označitvijo (n, k), pri čemer n predstavlja število vhodnih spremenljivk (vhodov), k pa število izhodnih spremenljivk opazovane funkcije. Na sliki 3.1 je ponazorjena osnovna shema reverzibilne logične funkcije.



Slika 3.1: Osnovna shema reverzibilne logične funkcije tipa (n,n).

Za primer pod drobnogled vzemimo nam znane enostavne logične funkcije identitete f_{ID} (funkcija tipa (1,1)), negacije f_{NEG} (funkcija tipa (1,1)) in dvovhodne konjunkcije f_{AND} (funkcija tipa (2,1)). Iz pravilnostnih tabel prvih dveh funkcij, predstavljenih v tabeli 3.1, lahko ugotovimo, da sta obe funkciji reverzibilni, saj lahko v obeh primerih na osnovi vektorja izhodnih vrednosti enolično določimo vektor vhodnih vrednosti. Iz podatkov v tabeli 3.2 lahko ugotovimo, da dvovhodna AND funkcija ni reverzibilna, saj na osnovi vektorja izhodne vrednosti ne moremo enolično določiti vektorja vhodnih vrednosti. Na nereverzibilnost funkcije lahko sklepamo že iz njenega tipa, ki je v tem primeru (2,1).

Do sedaj smo ugotovili, da logična funkcija tipa (n,k) prav gotovo ni reverzibilna, če sta naravni števili n in k različni ($n \neq k$). V primeru logične funkcije,

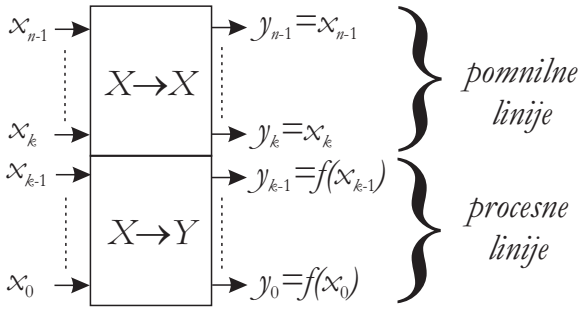
x_1	x_2	$f_{AND}(x)$
0	0	0
0	1	0
1	0	0
1	1	1

Tabela 3.2: Pravilnostna tabela dvovhodne nereverzibilne AND funkcije tipa (2,1).

ki ni reverzibilna, si lahko pomagamo z njenim dopolnjevanjem do reverzibilne logične funkcije. Pod dopolnjevanjem imamo v mislih bodisi dodajanje novih vhodnih spremenljivk, katere imenujemo za *konstantne vhode* (angl. *constant inputs*), ali pa dodajanje novih izhodov funkcije, katere imenujemo za *redundantne izhode* (angl. *garbage*) [24]. Dodajanje vhodnih spremenljivk in izhodov funkcije moramo izvesti tako, da zadostimo zahtevi po bijektivnosti preslikav vektorjev vhodnih vrednosti v vektorje izhodnih vrednosti. Ob upoštevanju tega pravila nas dopolnjevanje logične funkcije pripelje do funkcije tipa (n,n) in njene logične reverzibilnosti.

3.4 Osnovni model reverzibilnih logičnih funkcij

Do sedaj smo se v tekstu sklicevali na reverzibilnost logičnih funkcij. Na enak način se lahko sklicujemo na *reverzibilnost logičnih vezij*. Pod pojmom reverzibilnega logičnega vezja smatramo realizacijo neke reverzibilne logične funkcije. Reverzibilno logično vezje lahko ponazorimo z grafično predstavitvijo na sliki 3.2, kjer x_0, x_1, ...,x_{n-1} predstavljajo vhode, y_0, y_1, ...,y_{n-1} pa izhode reverzibilnega logičnega vezja. V splošnem za reverzibilna vezja velja, da se ob n



Slika 3.2: Osnovna shema reverzibilnega logičnega vezja tipa (n,n).

vhodnih vrednostih $n-k$ vhodnih vrednosti neposredno preslikuje na izhod, preostalih k vhodnih vrednosti pa se funkcijsko obdela. To je grafično predstavljeno na sliki 3.2, kjer se vhodne vrednosti x_k, x_{k+1}, ...,x_{n-1} neposredno

x	$f(x)$
0	y_0
1	y_1

Tabela 3.3: Splošna pravilnostna tabela funkcije tipa (1,1).

preslikujejo na izhod, vhodne vrednosti x_0, x_1, ...,x_{k-1} pa se pred posredovanjem na izhod funkcijsko obdelajo. Prve imenujemo za *pomnilne linije* (angl. *temporary storage*), druge pa za *procesne linije* (angl. *processing lines, target lines*) [24]. Na prvi pogled se zdi, da so pomnilne linije v vezju redundatne, čemur pa ni tako. V nadaljevanju pričujočega razdelka bomo ugotovili, da se vrednosti s pomnilnih linij ne preslikujejo samo na izhod, temveč posredujejo tudi v del, kjer prihaja do funkcijskih evaluacij (izvajanja funkcije f).

3.5 Reverzibilne logične funkcije tipa (1,1)

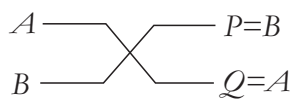
V tabeli 3.3 je prikazana splošna pravilnostna tabela funkcij tipa (1,1). Obstajajo 4 različne funkcije tipa (1,1) in sicer identiteta, negacija, konstanta 1 in konstanta 0. Ob zapisu pravilnostnih tabel vseh štirih funkcij bi lahko ugotovili, da sta prvi dve reverzibilni, obe funkciji posredovanja konstante na izhod pa nereverzibilni, saj v teh dveh primerih na osnovi izhodnih vrednosti ne moremo enolično določiti vhodnih.

3.6 Reverzibilne logične funkcije tipa (2,2)

Obstaja 256 (2^8) različnih logičnih funkcij tipa (2,2), pri čemer je število reverzibilnih v tem naboru dosti manjše (konkretneje 24 oziroma 4!). V domeni reverzibilnih funkcij tipa (2,2) bomo izpostavili Feynmanovo in SWAP reverzibilno logično funkcijo in njuni implementaciji z logičnimi vezji.

3.6.1 SWAP funkcija

Pomen SWAP funkcije je predstavljen s pravilnostno tabelo 3.4 in z grafično ponazoritvijo njenega logičnega vezja na sliki 3.3. Iz obeh je razvidno, da SWAP funkcija vrši zamenjavo vrstnega reda vhodnih vrednosti na izhodu in da je reverzibilna. Obe liniji ($A - Q$, $B - P$) sta pomnilnega značaja.



Slika 3.3: Shema reverzibilnega SWAP logičnega vezja.

A	B	P	Q
0	0	0	0
0	1	1	0
1	0	0	1
1	1	1	1

Tabela 3.4: Pravilnostna tabela SWAP reverzibilne logične funkcije tipa (2,2).

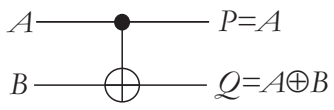
A	B	P	Q
0	0	0	0
0	1	0	1
1	0	1	1
1	1	1	0

Tabela 3.5: Pravilnostna tabela Feynmanove reverzibilne logične funkcije tipa (2,2).

3.6.2 Feynmanova funkcija

Ena od najbolj razširjenih reverzibilnih logičnih funkcij tipa (2,2) je Feynmanova funkcija. V angleškem jeziku jo poimenujemo tudi za *Controlled-NOT* (CNOT) funkcijo. Pomen Feynmanove funkcije je predstavljen s pravilnostno tabelo 3.5 in z grafično ponazoritvijo njenega logičnega vezja na sliki 3.4.

Iz grafične ponazoritve vezja je razvidno, da je linija $A-P$ sicer pomnilnega značaja, a istočasno vrednost vhoda A vstopa tudi v procesni del. Procesna linija $B-Q$ je realizirana z operatorjem XOR, ki ga v grafičnem smislu predstavlja znak $\oplus$. Iz pravilnostne tabele 3.5 je razvidno, da poimenovanje CNOT izvira iz kontrolirane negacije drugega vhoda B. Slednji se negira pod pogojem, da je kontrolni vhod $A = 1$. V primeru, da je kontrolni vhod $A = 0$, se B ne negira in s tem se vektor vhodnih vrednosti na izhodu ohranja. Istočasno lahko iz tabele razberemo, da v primeru vhodne vrednosti $B = 0$ funkcija vrši nalogo podvojevanja (angl. *fanout*) vhodnega signala A ($P = A$, $Q = A$). Klasična „fanout“ funkcija namreč zaradi neustrezanja zahtevi po tipu funkcije (n,n) sama po sebi ne sodi med reverzibilne logične primitive.



Slika 3.4: Shema reverzibilnega Feynmanovega (CNOT) logičnega vezja.

A	B	C	P	Q	R
0	0	0	0	0	0
0	0	1	0	0	1
0	1	0	0	1	0
0	1	1	0	1	1
1	0	0	1	0	0
1	0	1	1	0	1
1	1	0	1	1	1
1	1	1	1	1	0

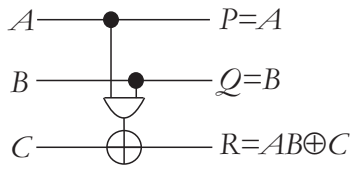
Tabela 3.6: Pravilnostna tabela Toffolijeve reverzibilne logične funkcije tipa (3,3).

3.7 Reverzibilne logične funkcije tipa (3,3)

Večina raziskav na področju reverzibilnih logičnih funkcij je usmerjena v funkcije tipa (3,3), saj naj bi le te predstavljale najboljšo osnovo za prehod na reverzibilno procesiranje. Vhodno izhodni prostor teh funkcij nam nudi $(2^8 * 2^8 * 2^8)$ 16.777.216 različnih logičnih funkcij. Od tega celotnega nabora različnih logičnih funkcij je 40.320 oziroma 8! logičnih funkcij reverzibilnih. V okviru funkcij tipa (3,3) bomo spoznali Toffolijevo in Fredkinovo reverzibilno funkcijo.

3.7.1 Toffolijeva funkcija

Toffolijevo reverzibilno funkcijo imenujemo tudi za *Controlled-controlled-not* funkcijo. Njena posebnost je, da predstavlja *poln funkcijski nabor*, kar pomeni, da poljubno logično funkcijo lahko realiziramo z množico Toffolijevih funkcij. V tabeli 3.6 je predstavljena pravilnostna tabela Toffolijeve funkcije, na sliki 3.5 pa grafična ponazoritev logičnega vezja te funkcije. Iz slike je razvidno, da ima vezje dve pomnilni (angl. *two through*) in eno procesno linijo. Dvojno



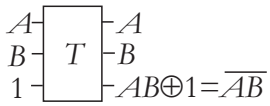
Slika 3.5: Shema logičnega vezja reverzibilne Toffolijeve logične funkcija tipa (3,3).

kontrolirano negiranje Toffolijeve funkcije se izkazuje skozi njeno procesno linijo. V primeru, da velja relacija $A = B = 1$, se izvede negacija vrednosti C $(R = neg(C))$, v primeru pa ko relacija $A = B = 1$ ne velja, do negacije ne pride in se na izhod prenaša vrednost C $(R = C)$.

A	B	C	P	Q	R
0	0	0	0	0	0
0	0	1	0	0	1
0	1	0	0	1	0
0	1	1	0	1	1
1	0	0	1	0	0
1	0	1	1	1	0
1	1	0	1	0	1
1	1	1	1	1	1

Tabela 3.7: Pravilnostna tabela Fredkinove reverzibilne logične funkcije tipa (3,3).

Na sliki 3.6 je prikazana realizacija logičnega vezja dvovhodnih NAND vrat s Toffolijevimi vrati. Ker dvovhodna NAND funkcija predstavlja poln funkcijski nabor, s slednjim dokažemo, da tudi Toffolijeva funkcija predstavlja poln funkcijski nabor.



Slika 3.6: Shema trovhodnega Toffolijevega logičnega vezja za izvedbo dvovhodne NAND logične operacije.

3.7.2 Fredkinova funkcija

Fredkinovo funkcijo tipa (3,3) imenujemo tudi za *Controlled SWAP* funkcijo. Funkcija je reverzibilna in predstavlja *poln funkcijski nabor*. Namesto s shemo logičnega vezja tokrat funkcijo ponazorimo z izrazi

$$P = A, \tag{3.5}$$

$$A = 1 : Q = C, R = B, \tag{3.6}$$

$$A = 0 : Q = B, R = C, \tag{3.7}$$

ali logičnimi izrazi

$$P = A, \tag{3.8}$$

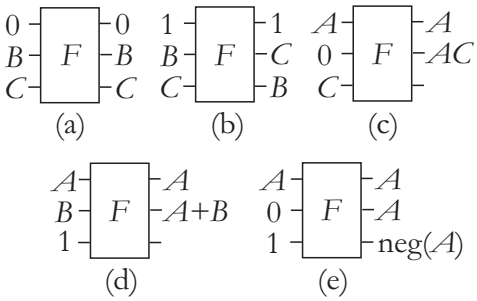
$$Q = B \oplus AB \oplus AC = \overline{A}B \oplus AC, \tag{3.9}$$

$$C = C \oplus AB \oplus AC = \overline{A}C \oplus AB. \tag{3.10}$$

V tabeli 3.7 je prikazana pravilnostna tabela Fredkinove funkcije. Iz pravilnostne tabele je razvidno, da je Fredkinova funkcija „one through“ funkcija ali funkcija

z eno pomnilno linijo. Kontroliranje zamenjave („Control SWAP“) vrednosti izhodov je pogojena z vrednostjo vhoda A. Ko je slednji po vrednosti 1 ($A = 1$), se na izhodih vrednosti vhodnih spremenljivk B in C zamenjata (preslikata po vrsti v vrednosti R in Q).

Na sliki 3.7 je prikazanih nekaj konfiguracij spremenljivih in konstantnih vhodov v Fredkinovo vezje, ki nam ponujajo realizacije uporabnih logičnih funkcij. Po vrsti so tako predstavljeni „three through“ vezje (a), SWAP vezje (b), AND vezje (c), OR vezje (d) in vezje negacije (e). Z realizacijo funkcijsko polnega nabora (AND, OR, NEG) smo potrdili tudi univerzalnost Fredkinove funkcije.



Slika 3.7: Uporaba trovhodnih Fredkinovih logičnih vezij za izvedbo enostavnih logičnih funkcij „three through“ (a), SWAP (b), AND (c), OR (d) in negacije (e).

3.8 Snovanje kompleksnejših reverzibilnih logičnih funkcijskih struktur

Gradnja kompleksnejših reverzibilnih logičnih funkcij ali njihovih realizacij v obliki reverzibilnih logičnih vezij (angl. *logic synthesis methods for reversible logic*) temelji na različnih metodah snovanja. Te so sledeče [24]:

- *kompozicijske metode*: temeljijo na uporabi osnovnih znanih reverzibilnih gradnikov (metoda konvencionalne logične sinteze);
- *dekompozicijske metode*: željeno funkcijo razbijemo na več podfunkcij, ki jih v nadaljevanju realiziramo;
- *metode genetskih algoritmov*: s simulacijo umetne evolucije skušamo doseči ustrezno solucijo (konfiguracijo entitet z ustreznim odzivom) v prostoru entitet;
- *preiskovalne metode*: s preiskovanjem prostora entitet skušamo na časovno potraten način najti ustrezno rešitev v množici različnih razporeditev osnovnih entitet;

V vseh metodah je prisotna ideja sinteznega pristopa k realizaciji reverzibilnega logičnega vezja.

3.9 Motivacija za vpeljavo reverzibilnosti procesiranja

Osnovna motivacija področja raziskav reverzibilnosti dinamike je iskanje hipotetičnih platform, ki bi bile zmožne reverzibilnega procesiranja nad diskretnim prostorom stanj. Slednje bi doprineslo predvsem k *energetsko efektivnejši* izrabi platforme. Rolf Landauer in John von Neumann [22] izgubo količine energije ob ireverziblini logični operaciji na bit definirata z njeno najmanjšo spodnjo mejo (limito). Minimalna izguba energije E se izračuna po izrazu

$$E \geq k * T * \ln 2, \tag{3.11}$$

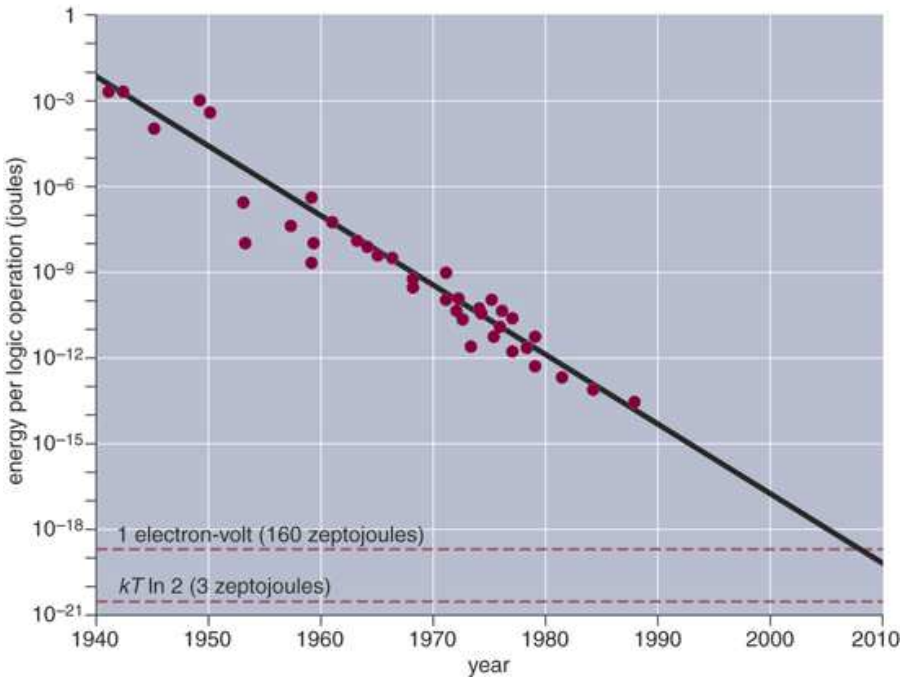
pri čemer je T temperatura procesnega medija, k pa Boltzmanova konstanta ($k = 1,38 * 10^{-23} J/K$). Pri sobni temperaturi (300 kelvinov) se vrednost E giblje v velikostnem razredu $3*10^{-21}$ J ali 3 zepto joulov. V večini primerov se izguba energije odraža v obliki *disipacije toplote*. Idealizirane sisteme, pri katerih ne prihaja do izmenjave toplote med sistemom in okoljem, imenujemo za *adiabatne sisteme*.

Pojem logične operacije nad bitom iz predhodnega odstavka si interpretiramo kot izbris bita. Prednost, ki jo doprinese reverzibilno procesiranje, je v brezizgubnem izvajanju logičnih operacij brez brisanja bitov. Tako naj bi reverzibilno procesiranje omogočalo poljubno zmanjševanje izgubljene energije, ki bi tako lahka padla tudi pod omenjeno mejno spodnjo vrednost E [25].

Na sliki 3.8 je predstavljen prikaz porabe energije na logično operacijo [22] (podatki glasijo na leto 1988). Von Neuman in Landauer predlagata za zmanjšanje limite dve metodi:

- procesiranje pri manjših temperaturah, kar je z vidika cene današnjih računalnikov in prehoda na vseprisotno računalništvo dandanes nesprejemljivo;
- prehod na reverzibilno procesiranje, ki bi omogočalo poljubno znižanje izgube energije pod mejo E;

V enem od predhodnjih razdelkov smo omenili, da dvovhodna dvovrednostna logična funkcija AND ni reverzibilna. Za to funkcijo velja, da potrebuje na vhodni strani večjo količino informacije, kot jo sprocesira na izhodni strani. Razlika omenjenih količin predstavlja informacijsko izgubo, kar se v današnjih elektronskih vezjih manifestira v disipaciji toplote. Pri reverzibilnih vratih, kjer naj do izgube informacij v večini primerov ne bi prihajalo, v večini primerov tudi ne bi prihajalo do izgube energije. Tako je osnovni cilj, ki ga skuša reverzibilnost doseči, *informacijsko brezizgubno procesiranje* (angl. *information lossless*) z razliko od današnjega informacijsko izgubnega, ko skozi procesiranje



Slika 3.8: Poraba energije na logično operacijo [22].

dvovhodnih operacij praktično na vsakem koraku procesiranja izgubljamo (brišemo) informacijsko vsebino. Reverzibilnost naj bi omogočala, da se nobena informacija o minulih stanjih skozi čas ne bi izgubila. Na ta način lahko dosežemo kakršnokoli zgodovinsko stanje sistema z vzvratnim procesiranjem (angl. *computing backwards*, *un-computing the results*) [23].

3.10 Povzetek poglavja

V devetdesetih letih prejšnjega stoletja je prišlo kar do nekaj realizacij procesorjev, ki procesirajo zgolj na osnovi reverzibilnih logičnih funkcij. Eden od najvidnejših avtorjev na tem področju je M. Frank [26].

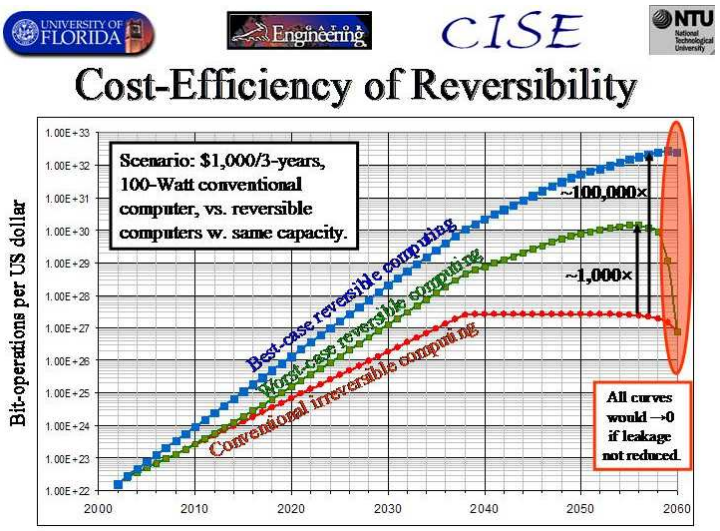
Temeljna motivacija za reverzibilno procesiranje izhaja iz iskanja energetsko optimalnejše procesne strukture. Seveda se s tem poraja kar nekaj problemov. Naj jih naštejemo le nekaj:

- Kakšna je materialna platforma kot osnova za procesiranje, na kateri bi se lahko spustili pod Landauerjevo limito (danes smo še za nekaj velikostnih razredov nad njo)?
- Kaj je z obrnljivostjo izgubnih procesov zaokroževanja podatkov v procesu

digitalizacije (npr. $\frac{1}{3} \rightarrow 0,33333$)?

- Kaj je z obrnljivostjo sekvence prevajanja izvorna koda $\rightarrow$ eksekucijska koda?

Poleg same procesne alternativne platforme, ki bi takšno procesiranje omogočala, se v prihodnosti velik pomen pripisuje tudi prevajalnikom, ki naj bi pri prevajanju maksimizirali število reverzibilnih funkcij. Na sliki 3.9 je predstavljena ocena cenovne efektivnosti različnih vrst procesiranja [27].



Slika 3.9: Cenovna efektivnost glede na različne tipe procesiranja [27].

V zadnjem desetletju se raziskave o možnostih uporabe reverzibilnosti intenzivirajo. Tako so npr. razviti že prvi programski jeziki, za podporo reverzibilnosti (primer takšnega jezika je JANUS), analizatorji reverzibilnih vezij (RCViewer+) itd. Več o osnovah reverzibilnega procesiranja bralec najde v delu [28].

Literatura

[1] "The scale of things." `http://science.energy.gov/bes/community-resources/scale-of-things-chart/`, September 2016.

[2] M. Hak, *The MEMS Handbook*. CRC Press, 2002.

[3] M. Mack, "The multiple lives of Moore's law," *IEEE Spectrum*, vol. 4, pp. 29–35, 2015.

[4] D. Kodek, *Arhitektura in organizacija računalniških sistemov*. Bi-Tim, Slovenija, 2008.

[5] *From editors of Scientific American: Understanding nanotechnology*. Warner Books, ZDA, 2002.

[6] "2001: A Space Odyssey," 1968.

[7] A. Adamatzky, B. Costello, and T. Asai, *Reaction diffusion computers*. Elsevier, 2005.

[8] B. Hayes, "Third base," *American Scientist*, vol. 89, no. 6, 2001.

[9] W. Aspray, *John Von Neumann and The Origins Of Modern Computing*. The MIT Press, England, 1990.

[10] "There's Plenty of Room at the Bottom." `https://en.wikipedia.org/wiki/There%27s_Plenty_of_Room_at_the_Bottom/`, September 2016.

[11] E. Regis, *Nano – the emerging science of nanotechnology*. BackBay Books, 1995.

[12] C. Lent, P. Tougaw, W. Porod, and G. Bernstein, "Quantum cellular automata," *Nanotechnology*, vol. 4, 1993.

[13] C. Lent and P. Tougaw, "Lines of interacting quantum-dot cells: a binary wire," *Journal of Applied Physics*, vol. 74, 1993.

[14] I. L. Bajec and M. Mraz, "Večstanjsko procesiranje v strukturah kvantnih celičnih avtomatov," *Elektrotehniški vestnik*, vol. 73, no. 2-3, 2006.

[15] P. Pečar, "Uporaba adiabatnega pristopa pri realizaciji trojiškega procesiranja na osnovi kvantnih celičnih avtomatov," Master's thesis, Faculty of computer and Information science, University of Ljubljana, 2007.

[16] G. Snider, A. Orlov, I. Amlani, and G. Bernstein, "Quantum-dot cellular automata: line and majority logic gate," *Japanese Journal of Applied Physics*, vol. 38, 1999.

[17] K. Walus, T. Dysart, G. Jullien, and R. Budiman, "Qcadesigner: a rapid design and simulation tool for quantum dots cellular automata," *IEEE Transactions on Nanotechnology*, vol. 3, 2004.

[18] T. Cole and J. Lusth, "Quantum-dot cellular automata," *Progress in Quantum Electronics*, vol. 25, 2001.

[19] Z. Kohavi, *Switching and finite automata theory.* McGraw-Hill Inc., USA, 1978.

[20] M. Niemier and P. Kogge, *Nano, Quantum and Molecular Computing - Implications to High Level Design and Validation*, ch. Origins and Motivations for Design Rules in QCA. Kluwer Academic Publishers, Boston, 2004.

[21] "Reverzibilnost procesiranja." `http://strangepaths.com/reversible-computation/2008/01/20/en/`, Oktober 2016.

[22] B. Hayes, "Reverse engineering," *American Scientist*, vol. 94, 2006.

[23] P. R. Yelekar and S. S. C. Hanson, "Introduction to reversible logic gates and its application," *International journal of computer applications*, 2011.

[24] S. M. R. Taha, *Reversible logic synthesis methodologies with application to quantum computing.* Springer, Switzerland, 2016.

[25] P. J. Denning and T. G. Lewis, "Computers that can run backwards," *American Scientist*, vol. 105, no. 5, 2017.

[26] "Reversible." `http://www.eng.fsu.edu/\symbol{126}mpf/pubs.htm`, Maj 2015.

[27] "Reversible." `http://www.cise.ufl.edu/research/revcomp/`, Maj 2015.

[28] K. Perumalla, *Introduction to Reversible Computing.* Chapman & Hall/CRC Press, 2014.