

Ilkka Virtanen

6. Esimerkki heuristisen menetelmän avulla peliä pelaavasta ohjelmasta

Tietokoneella pelattavista peleistä yleensä

Suurin piirtein kai niin kauan kuin tietokoneet ovat olleet olemassa, on näillä yritetty ratkaista erilaisia pelejä, ts. saada tietokone (tai oikeammin koneeseen syötetty ohjelma) pelaamaan niin hyvin, että se voittaisi minkä tahansa inhimillisen vastustajan. Suurimman mielenkiinnon kohteena on luonnollisesti ollut shakki, valkeutensa ja laajan suosionsa takia. Shakinpeluohjelmat (kts. [1], ss. 33-61) ovat nykyisellään kehittyneet niin pitkälle, että ne pystyvät menestyksekkäästi pelaamaan keskitasoisia, vieläpä hyviäkin pelaajia vastaan, mutta ovat sensijaan voimattomia mestareita vastaan.

Tietokoneen suorittamaa pelaamista on tutkittu osaksi huviksi, osaksi sen arvon vuoksi, mikä sillä on talous- ja yhteiskuntaelämän päätöksenteon mallina. Pelien pelaamisessa käytettäviä menetelmiä voidaan käyttää varsinkin tilanteissa, joissa probleemat ovat hyvin määriteltäviä, mutta kuitenkin niin monimutkaisia, että äly ja päättelykyky, usein myös kokemus ja intuitio ovat tarpeen.

Ryhdyttäessä ohjelmoimaan koneelle jotakin peliä on valittavana useitakin eri teitä, joita pitkin voidaan edetä. Suoraviivaisin vaihtoehto, joka samalla merkitsee absoluuttisen hyvää pelaamista, on menettely, jossa jokaisen siirtovuoron yhteydessä tutkitaan kaikki siirtovaihtoehdot, näihin liittyvät vastasiirtomahdollisuudet jne. kunnes päästään tilanteeseen, jossa peli on ratkennut. Menetelmä johtaa ehdottomaan optimiin ja on erinomainen hyvin yksinkertaisten pelien kyseessä ollessa. Sen sijaan jo vähänkin vaikeammassa peleissä kaikkien mahdollisten siirtokombinaatioiden lukumäärä tulee niin suureksi, että niiden tutkiminen on liikaa aikaa vaativaa tai suorastaan mahdotonta. Esimerkiksi shakissa on erilaisten pelitilanteiden lukumäärä suuruusluokkaa 10^{120} . (Vertailun vuoksi mainittakoon, että vuosisadassa on alle 10^{16} mikrosekuntia). Tästä esimerkistä jo voidaan päätellä, että minkään

TURUN
KAUPPAKORKEAKOULUN
JULKAISUJA
SARJA

PUBLICATIONS OF THE TURKU
SCHOOL OF ECONOMICS
SERIES

Ilkka Virtanen - Pekka Aho - Lasse Kurkilahti -
Anja Leinokari - Paavo Mattila - Tuula Vilja:

HEURISTINEN OHJELMOINTI

Eripainos julkaisusta: MIS-70 SEMINAARITYÖT (Turun Kauppa-
korkeakoulun julkaisuja C I - 1 : 1970, osa III, ss. 56-113)

todella mielenkiintoisen pelin ratkaisemiseen edellä esitetty menetelmä ei sovi.

Valittaviksi vaihtoehdoiksi jäävät siis menetelmät, joissa kaikkien mahdollisten siirtokombinaatioiden sijasta tutkitaan huomattavasti pienempää joukkoa siirtomahdollisuuksia ja näiden perusteella tehdään päätös. Tällaiset menetelmät ovat useimmiten luonteeltaan heuristisia, sillä tutkittavien vaihtoehtojen valinnassa on erilaisilla heuristiikoilla huomattava merkitys. Ainakin seuraavan tyyppisiä ohjelmia on laadittu eri pelejä varten.

1. Oppivat ohjelmat. Peliä pelattaessa ohjelma taltioi kuhunkin ratkaisuun liittyvät seuraamukset niin, että kun peliä on pelattu tarpeeksi paljon, ohjelma muistaa kuhunkin pelitilanteeseen liittyvät hyvät ja huonot siirrot.

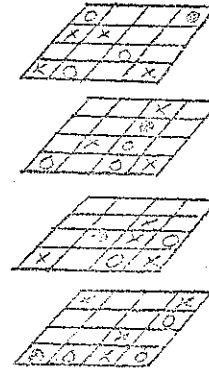
2. Siirtosarjojen tutkiminen tiettyyn "syvyyteen" asti. Pelitilanteessa tutkitaan kaikki mahdolliset siirtovaihtoehdot, näihin kaikki mahdolliset vastasiirrot jne. esimerkiksi viiden vuoroparin syvyyteen asti. Siirtöpäätös tehdään tänä aikana syntyvien tilanteiden perusteella.

3. "Branch and bound"-menetelmät. Näissä menetelmissä on joidenkin sääntöjen avulla mahdollista määrittää vaihtoehdot, jotka eivät missään tapauksessa voi tulla kysymykseen. Näin voidaan suurin osa vaihtoehdoista jättää tarkastelun ulkopuolelle ja päätös tehdä suhteellisen suppean joukon keskuudessa.

4. Muutamien kriittisten tilanteiden tutkiminen. Laaditaan luettelo tilanteista, joissa jospikumpi pelaajista on voittamassa tai ainakin muutamalla siirrolla voi päästä tällaiseen vaiheeseen. Aina ennen siirtoa tutkitaan esiintyykö pelilaudalla jokin luettelossa mainituista tilanteista. Mikäli esiintyy, tehdään tilanteen vaatima siirto, muulloin voidaan siirtää miten tahansa. Seuraavassa esitettävä peli on ohjelmoitu tällä periaatteella.

Kolmiulotteinen "erämiehen shakki" (tic - tac - toe)

Kolmiulotteinen "erämiehen shakki" (tic - tac - toe, seuraavassa käytetään lyhennystä TTT3D), on vanha itämainen peli, jota pelataan nelitasoisella pelipöydällä. Kussakin tasossa on allekkain olevat 4 x 4 -ruudut (kuva 1).



kuva 1.

Kumpikin pelaaja varaa vuoronperään yhden ruudun omalla merkillään. Voittaja on se, joka ensin saa neljä omaa merkkiään peräkkäin samaan riviin. Rivi voi olla vaakatai pystytasossa tai se voi olla tasolta toiselle kulkeva lävistäjäriivi. Kuvan 1 esittämässä tilanteessa merkkiä O käyttänyt pelaaja on voittanut (voittorivin ympyrät mustattu). Rivejä on mahdollista muodostaa kaikkiaan 76:lla eri tavalla. Eräät ruudut pelipöydällä ovat muita "arvokkaampia", sillä ne esiintyvät muita useammassa rivissä. Tasojen 1 ja 4 kuimaruudut sekä tasojen 2 ja 3 keskiruudut (yhteensä 16 ruutua) ovat tällaisia, sillä ne esiintyvät kukin seitsemässä eri rivissä kun taas muut 48 ruutua esiintyvät kukin vain neljässä eri rivissä.

Pelipöydän, tilanteiden ja siirtojen kuvaus koneelle

Seuraavassa kuvataan R. K. Loudenin (kts. [2], ss. 179-204) esittämää FURIKAN-ohjelmaa TTT3D-pelin pelaamiseksi IBM 1130-tietokoneella. Tarkastellaan aluksi, miten koneelle voidaan kuvata pelipöytä, siirrot sekä pelipöydällä vallitseva tilanne. Ruudut on numeroitu 1-64. Pelipöytää kuvaa vektori MOVE (64), joka ilmoittaa kussakin ruudussa vallitsevan tilanteen. Jos

$$\text{MOVE (I)} = \begin{cases} 0, & \text{on ruutu I ryhjä} \\ 1, & \text{on ruudussa I vastustajan merkki,} \\ 5, & \text{on ruudussa I koneen merkki.} \end{cases}$$

Vektoriin SUM (76) on talletettu jokaisen 76:n rivin rivisummat (= rivien kuuluvia ruutuja vastaavien MOVE-vektorin komponenttien summat). Näin on saatu yksikäsitteinen vastaavuus rivillä vallit-

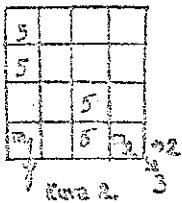
sevan tilanteen ja rivisumman välillä. Jos esimerkiksi SUM (10) = 11, merkitsee tämä, että rivillä 10 on kaksi koneen varaamaa ruutua, yksi vastustajan ruutu ja yksi tyhjä ruutu.

Kuhunkin riviin sisältyvien ruutujen indeksit on talletettu vektoriin ROW (304). Ruutujen indeksit annetaan koneelle ja kone tulostaa ne kolminumeroisena lukuna, jonka ensimmäinen numero ilmaisee tason, toinen rivin ja kolmas palstan. Siten esim. luku 312 tarkoittaa kolmannen tason ensimmäisen rivin toista ruutua. Koneelle tässä muodossa annettu luku muutetaan koneen suorittamia operaatioita varten välillä 1-64 olevaksi indeksiksi. Vastaavasti kone siirron tehtyään ilmoittaa ruudun sijainnin vastustajan koodin mukaisesti.

Ohjelman perustana oleva strategia

Kuten jo edellä viitattiin, strategian perustana on joukko tilanteita, joiden esiintymistä pelilaudalla tarkkailaan. Kaikkiaan 13 erilaisesta tilannetta tutkitaan määrättyssä järjestyksessä ja ensimmäinen tilanne, jonka havaitaan pelipöydällä esiintyvän, aiheuttaa tilanteen mukaisen toimenpiteen.

Esimerkkinä mainittakoon tilanne, jolloin kone havaitsee jonkin rivin summan olevan neljä. Tällöin vastustaja on voittanut pelin, joten koneen tehtäväksi jätetään ilmoituksen teko tästä voitosta. Rivisumma 15 taas merkitsee, että koneella on kolme merkkiä jossakin rivissä ja rivin neljäs ruutu on tyhjä. Siirtämällä tähän ruutuun kone voittaa pelin. Tarkastellaan vielä jo melkoisen monimutkaisista tilannetta. Olkoon rivillä 1, jonka summa = 10, tyhjä ruutu m_1 yhteisenä rivin 2 kanssa, jonka summa = 5, ja tällä toisella rivillä puolestaan yhteinen tyhjä ruutu m_2 rivin 3 kanssa, jonka summa = 10. (kuva 2)



Siirtämällä ruutuun m_1 kone pakottaa vastustajan siirtämään rivin 1 ainoaan vapaaseen ruutuun. Kun kone tämän jälkeen tekee siirron m_2 , on sekin rivissä 2 että rivissä 3 kolme koneen merkkiä. Vastustaja voi vuorollaan sulkea näistä vain toisen rivin, joten kone on voittaja.

Kaikkien 13 tilanteen etsiminen voidaan tehdä täysin saman kaavan mukaan ottamalla käyttöön ns. testiluvut t_1 , t_2 ja t_3 . Kullakin tilanteella on omat testilukunsa, joita käyttäen menetelmään seuraavasti:

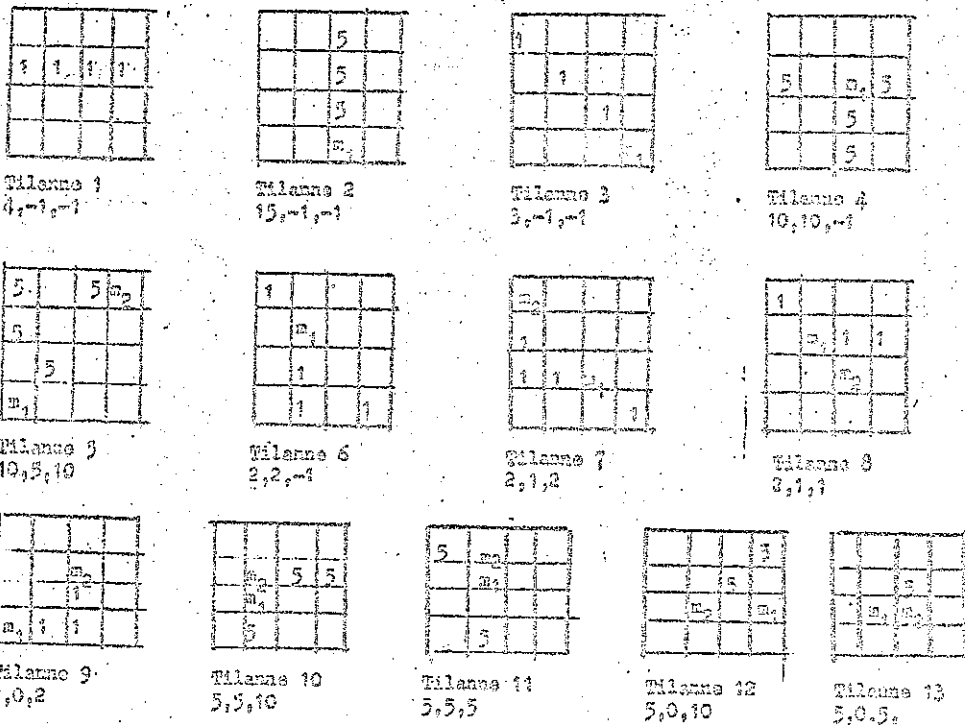
Etsitään pelipöydältä rivi, jonka summa = t_1 , siten että tällä rivillä on tyhjä ruutu m_1 yhteisenä toisen rivin kanssa, jonka summa = t_2 . Siten että tällä toisella rivillä on puolestaan tyhjä ruutu m_2 yhteisenä kolmannen rivin kanssa, jonka summa = t_3 . Ruudut m_1 ja m_2 voivat olla sama ruutu. Koneen seuraava siirto on ruutuun m_1 .

Yksinkertaisetkin tilanteet (kuten summa = 3) voidaan ilmaista testilukujen avulla käyttämällä negatiivisia lukuja. Ohjelma on laadittu siten, että negatiivisten testilukujen yli hypätään. Siten esimerkiksi testiluvut 3, -1, -1 aikaansaavat vain sen, että etsitään rivi, jonka summa = 3, ja tästä rivistä tyhjä ruutu.

Kuvassa 3 on esitetty kaikki 13 tutkittavaa tilannetta ja niihin liittyvät testiluvut. Tilanteet ovat siinä järjestyksessä kuin ne ohjelmassa läpi käydään. Piirroksat kuvaavat yksinkertaisuuden vuoksi kussakin tapauksessa vain tasotilannetta. Ohjelma luonnollisesti tutkii kaikki mahdolliset rivit.

Mikäli mikään tilanteista 1-13 ei esiinny pelipöydällä, siirrytään tutkimaan ns. odotuslistaa (vektori WAIT). Tämä sisältää strategisesti tärkeimpien ruutujen (em. 16 ruutua) indeksit. Näistä ensiksi löydettyyn vapaaseen ruutuun tehdään siirto. Mikäli kaikki 16 ruutua ovat varatut, peli päättyy tasapeliin, sillä kummallakaan pelaajalla ei ole enää mahdollisuutta saada riviään täyteen.

Vaikkakaan ei voida olla ehdottoman varmoja siitä, että edellä esitettyjen tilanteiden luettelo olisi optimaalinen tai että tilanteiden tutkimisjärjestys olisi paras mahdollinen, on osoittautunut, että kone pelaa voitokkaasti, vastustajan on erittäin vaikea päästä edes tasapeliin. Kone tekee siirron yleensä melko nopeasti; niissä tapauksissa, jolloin mikään tilanteista 1-13 ei esiinny pelilaudalla, siirron odottaminen voi kestää kuitenkin aina 30 sekuntiin saakka.



Kuva 3

TTT3D-ohjelman rakenne ja ominaisuudet

TTT3D-peliä pelaava ohjelma on laadittu niin, että pääohjelma, nimeltään TTT3D, huolehtii input - output - toiminnasta ja pelialueella vallitsevasta tilanteesta, STRAT-ali ohjelma suorittaa siirrot ja BOARD- ali ohjelma muuntaa siirtoalueen koodin vastustajan käyttämää muotoon. TTT3D-pääohjelma suorittaa mm. seuraavat toiminnot:

- lukee datat reikäkorteilta
- kuvaa peliä vastustajalle
- selostaa peliä tarkemmin niin haluttaessa

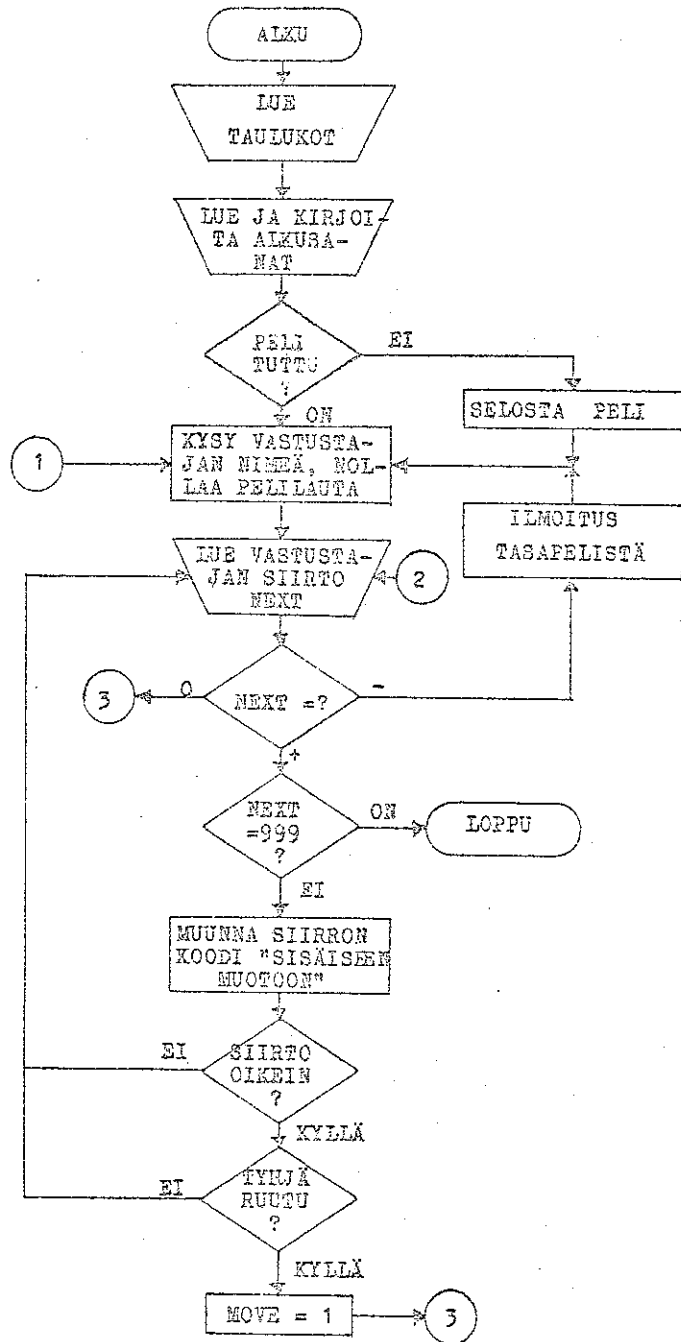
- kysyy vastustajan nimeä ja käyttää sitä "keskustelun" myöhemmissä vaiheissa
- antaa vastustajan valita pelin aloittajan
- kirjaa vastustajan siirrot ja ilmoittaa koneen siirrot (koneen siirron yhteydessä ilmoitetaan myös ruutu m_2 ja tilanteen numero)
- tarkistaa vastustajan siirron laillisuuden, hylkää virheellisen siirron ja tekee tästä ilmoituksen
- antaa ilmoituksen pelin päättymisestä (voitto, tappio, tasapeli)
- antaa vastustajalle mahdollisuuden siirtyä milloin tahansa uuteen peliin tai lopettaa pelaamisen.

Seuraavilla sivuilla olevissa liitteissä 1 a - 1 d on lohkokaavioesitys TTT3D-pääohjelmasta ja STRAT-ali ohjelmasta, liitteissä 2 a - 2 c listaus kolmen ohjelman muodostamasta ohjelmakokonaisuudesta, liitteessä 3 luettelo tarvittavista datakorteista ja liitteessä 4 esimerkki pelatusta pelistä.

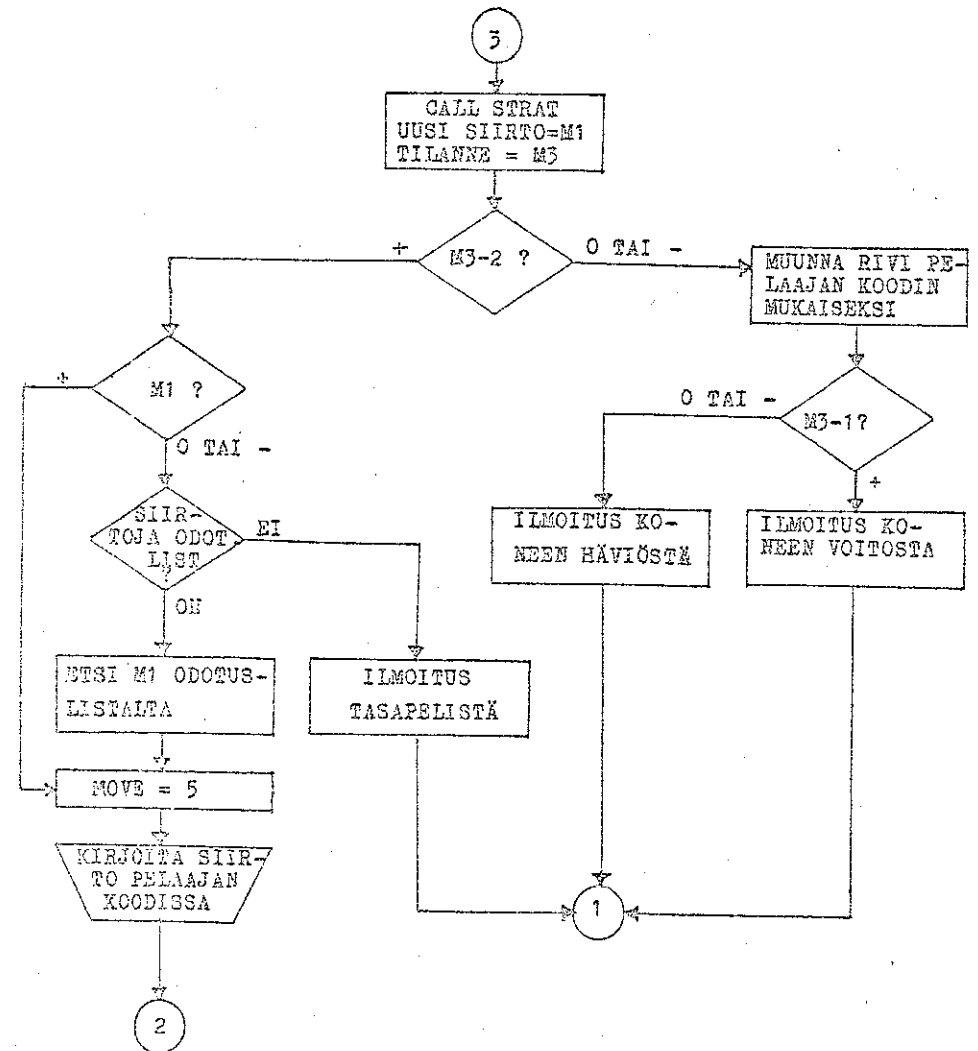
Kirjallisuus:

- 1) Tekoäly, Otadata - julkaisu c 5/1969, Otaniemi 1969.
- 2) R. K. Loudon: Programming the IBM 1130 and 1800, Englewood Cliffs 1967.

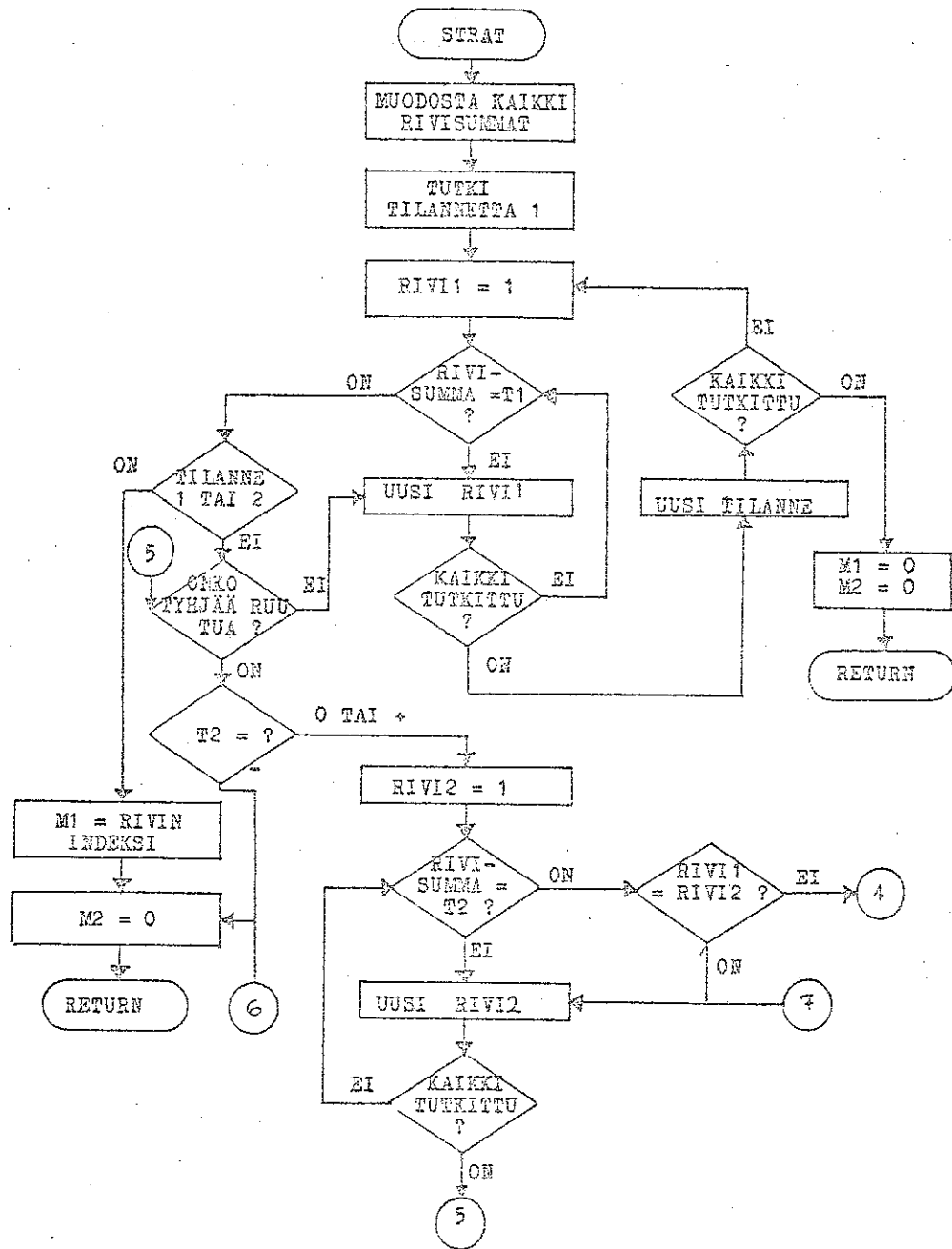
Liite 1a. TTT3D-ohjelman lohkokaavioesitys



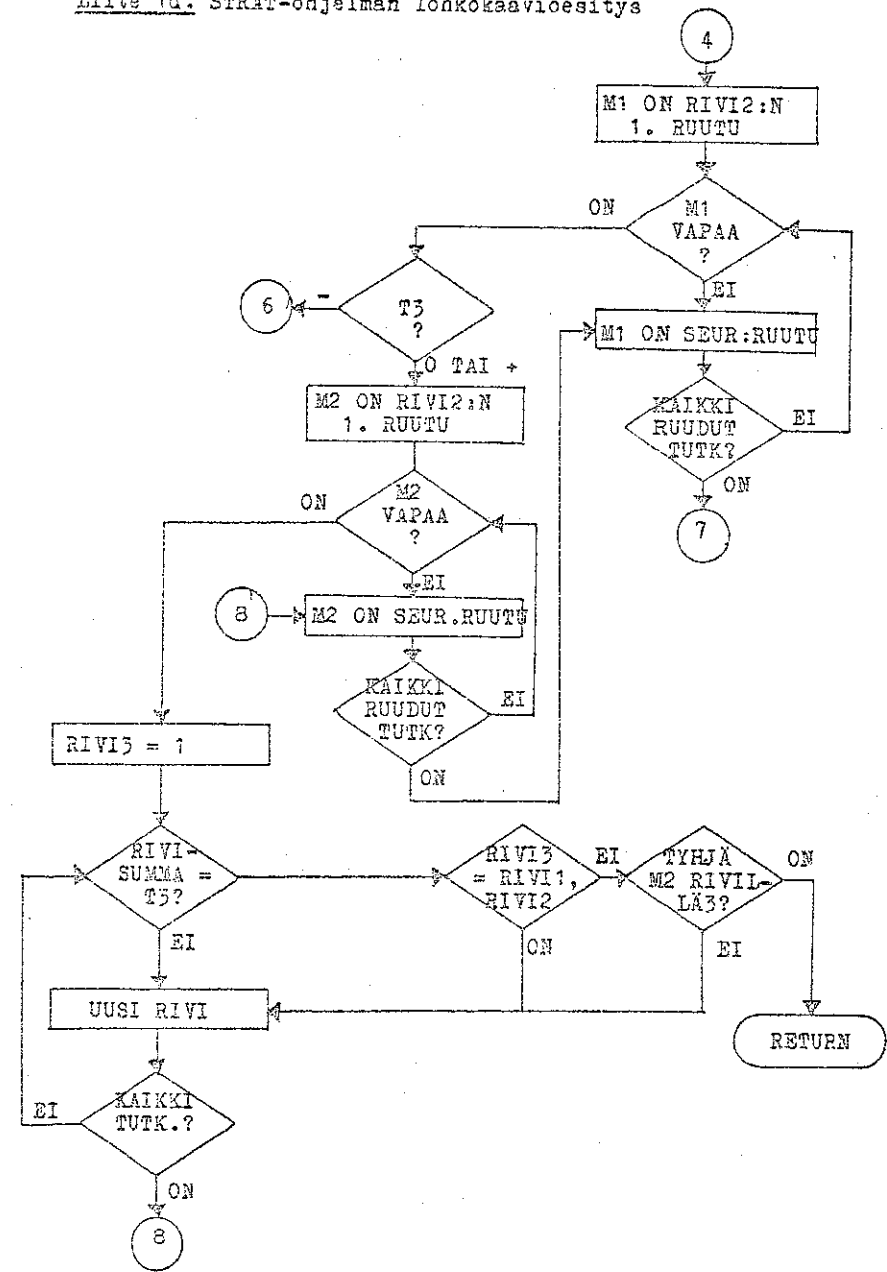
Liite 1b. TTT3D-ohjelman lohkokaavioesitys



Liite 1c. STRAT-ohjelman lohkokaavioesitys



Liite 1d. STRAT-ohjelman lohkokaavioesitys



// JOB T

LOG DRIVE CART SPEC CART AVAIL PHY DRIVE
0000 0008 0018 0000
6240 0001

V2 M06 ACTUAL 16K CONFIG 16K

// * TY6 TTT30 ERAMIEHEN SHAKKI, ILKKA VIRTANEN 8.5.1970.

// FOR

*ONE WORD INTEGERS

*LIST SOURCE PROGRAM

SUBROUTINE STRAT

INTEGER SUM(76),ROW(80),WAIT(20),TEST(2,4),TEST1,TEST2,TEST3

DIMENSION MOVE(64)

COMMON MOVE,SUM,ROW,WAIT,TEST,M1,M2,M3

C-----LASKETAAN ALUUKSI RIVIN RIVISUMMA

DO 15 J1 A 1,76

SUM(J1) A 0

K2 A 4*J1

L2 A K2-3

DO 15 J2 A L2,K2

J3 A ROW(J2)

15. SUM(J1) A SUM(J1)+MOVE(J2)

C-----TESTATAAN ENSIMMAISTA TILANNETTA

DO 21 J1 A 1,14

K2 A J1

TEST1 A TEST(1,J1)

IF(TEST1) 21,2,2

2. TEST2 A TEST(2,J1)

TEST3 A TEST(3,J1)

C-----ETSITAA RIVI, JONKA SUMMA ON TEST1

DO 22 J2 A 1,76

IF(SUM(J2)-TEST1) 22,3,22

3. IF(J1-2) 18,18,4

C-----ETSITAA TYHJA RUUTU RIVILTÄ, JONKA SUMMA ON TEST1

DO 23 J3 A 5,J2

L3 A K2-3

DO 23 J5 A L3,K2

M1 A ROW(J3)

IF(MOVE(M1)) 23,5,23

5. IF(TEST2) 17,10,10

10. ETSITAA RIVI, JONKA SUMMA ON TEST2

DO 24 J4 A 1,76

IF(SUM(J4)-TEST2) 24,6,24

6. IF(J3-2) 7,2,4

C-----ETSITAA TYHJA RUUTU M1 RIVILTÄ, JONKA SUMMA ON TEST2

DO 25 J5 A 4,J4

L5 A K5-3

DO 25 J5 A L5,K5

IF(M1-ROW(J5)) 25,8,25

8. IF(TEST3) 17,9,9

C-----ETSITÄ TYHJA RUUTU M2 RIVILTÄ, JONKA SUMMA OLI TEST2

DO 26 J6 A L5,K5

M2 A ROW(J6)

IF(MOVE(M2)) 26,11,26

C-----ETSITÄ RIVI, JONKA SUMMA ON TEST3

DO 27 J7 A 1,76

IF(SUM(J7)-TEST3) 27,12,27

12. IF(J7-J2) 13,27,13

13. IF(J7-J4) 13,27,13

C-----ETSITÄ TYHJA RUUTU M3 RIVILTÄ, JONKA SUMMA ON TEST3

DO 28 J8 A 4,J7

L8 A K8-3

DO 28 J8 A L8,K8

IF(M2-ROW(J8)) 28,16,28

28. CONTINUE

27. CONTINUE

26. CONTINUE

25. CONTINUE

24. CONTINUE

23. CONTINUE

22. CONTINUE

21. CONTINUE

C-----TÄNNE JOUDUTTAESSA MINÄN TILANNETTA -- 10 TI. ESTINNY

M1 A 0

GO TO 17

C-----TILANNE ON 1 TAI 2, M1 ILMAISEE RIVIN INDEKSI

18. M1 A J2

17. M2 A 0

16. RETURN

END

8/5 2 Lite 2b. FORTRAN-listaus

END OF COMPILATION

// DUP

*STORE WS UA STRAT

CART ID 0008 DB ADDR 29A3 DB CNT 0019

// FOR

*ONE WORD INTEGERS

*LIST SOURCE PROGRAM

SUBROUTINE BOARD1,NEWMV

C-----ALOHJELMA MUUNTAÄ SIIRTORUUDUN INDEKSIIN KONEEN KÄYTTÄMÄSTÄ MUO-

C-----DOSTA PELAAJAN KÄYTTÄMÄÄN MUOTOON

1. IF(I) 1,1,2

NEWMV A 1

2. RETURN

L1 A (I-1)/16

L2 A I-16*L1

L3 A (L3-1)/4

NEWMV A 100*(L1+1)+10*(L3+1)+L2-4*L3

RETURN

END

FEATURES SUPPORTED

ONE WORD INTEGERS

CORE REQUIREMENTS FOR BOARD

COMMON C VARIABLES 8 PROGRAM 90

END OF COMPILATION

// DUP

*STORE WS UA BOARD

CART ID 0009 DB ADDR 398C DB CNT 0006

// FOR

*NAME TTT30

*IOCARD,KEYBOARD,TYPEWRITER)

*ONE WORD INTEGERS

*LIST SOURCE PROGRAM

INTEGER SUM(76),ROW(80),WAIT(20),TEST(3,14),YE,ANS,BLANK

DIMENSION MOVE(64),LAST(4),NAME(10)

COMMON MOVE,SUM,ROW,WAIT,TEST,M1,M2,M3

89. FORMAT(54H

90. FORMAT(24I3)

91. FORMAT(15H VOITIT RIVILLÄ,4I4,2H, ,10A2)

92. FORMAT(14H VOITIN SINUT ,10A2,8H RIVILLÄ,4I4)

93. FORMAT(12H KATSOHAN TÄRKEÄÄ, ,10A2,20H, SIIRTOSI ON LAITON)

94. FORMAT(116H SINUN VUOROSI, ,10A2)

95. FORMAT(120H ILMOITÄKO NIMESI, OLE HYVÄ)

96. FORMAT(133H TASAPELI, ALOITETAÄN UDESTÄÄN, ,10A2)

97. FORMAT(10A2)

98. FORMAT(112H SIIRTOSI ON,14,12H STRATEGIANA,14,13)

99. FORMAT(15H SIIRTOSI OLI,14)

NTYP A 1

NCARD A 2

KEYBD A 6

C-----ASETAÄN ODOTUSLISTAN INDEKSILLE K1 ALKUARVO 1

K1 A 1

C-----LUETAÄN DATATAULUKOT

READ(INCARD,90) ROW,WAIT,TEST

READ(INCARD,91) YE,BLANK

C-----LUETAÄN JA KIRJOITETAÄN PELIN ALKULAUSEET

DO 5 I A 1,5

READ(INCARD,89)

WRITE(NTYP,89)

C-----TUTKITAÄN, TUNTEKO PELAAJA PELIN SAÄNNÖT

READ(KEYBD,97) ANS

C-----SELOSTETAÄN PELISAÄNNÖT TARVITTAESSA

DO 10 I A 1,10

READ(INCARD,89)

IF(YE-ANS) 10,21,10

21. WRITE(NTYP,89)

10. CONTINUE

C-----KYSYÄÄN VÄSTUSTAJÄN NIMEÄ

DO 20 I A 1,10

4. NAME(I) A BLANK

20. WRITE(NTYP,95)

READ(KEYBD,97) NAME

C-----MOLLÄTÄÄN PELILÄUTÄ

DO 1 I A 1,64

MOVE(I) A 0

C-----KIRJÄTÄÄN VÄSTUSTAJÄN SIIRTO, TÄRKISTETAÄN HALUÄÄKO HÄÄN ANTÄÄ

C-----ALKUSIIRTOÄ KONEELLE TÄI HALUÄÄKO HÄÄN MÄHDÖLLISESTI KESKEYTTÄÄ

C-----KÄYTTÄESSÄ OLEVÄÄN PELIN

WRITE(NTYP,94) NAME

BRAYTBOARD 16K

```

WRITE (NTYP,90) NEXT
IF (NEXT) 19,3,6
C----- PELI SEIS JA PILLIT PUSSIIN
9  IF (NEXT=999) 29,28,28
28 CALL EXIT
C----- SUUNNITTAAN SIIRRON KOODIA JA TARKISTETAAN SIIRRON LAILLISUUS
29 K1 A NEXT/100
K3 A NEXT-100*K1
K2 A K3/10
Y3 A K3-10*K2
IF (K1) 8,8,22
22 IF (K1-4) 23,23,8
23 IF (K2) 5,5,24
24 IF (K2-4) 25,25,8
25 IF (K3) 8,8,26
26 IF (K3-4) 7,7,8
7 NEXT A 16*(K1-1)+4*(K2-1)+K3
C----- TARKISTETAAN, ETTA VASTUSTAJAN SIIRTO TAPAHTUI TYHJAAN RUUTUUN
IF (MOVE (NEXT)) 9,9,8
8 WRITE (NTYP,93) NAME
GO TO 2
C----- VASTUSTAJAN VARAAMA RUUTU MERKITÄÄN 1-LLÄ
9 MOVE (NEXT) A 1
C----- KUTSUTAAN ALI OHJELMAA STRAT TILANNEARVIOINTIA JA SIIRRON SUORIT-
C----- TAMISTA VARTEN. M1 ON PARAS SIIRTO, M2 VAIHTOEHTO, M3 ILMOITTAA
C----- TILANTEEN, JOSSA OLLAAN
3 CALL STRAT
C----- TUTKITAAN, ONKO TILANNE 1 TAI 2, TÄLLÖIN PELI PÄÄTTY
17 IF (M3=2) 17,17,18
K3 A 4*(M1-1)
DO 11 I A 1+4
K2 A I+K2
11 CALL BOARD (ROW(K2),LAST(I))
IF (M3=1) 15,15,16
C----- KONE ON HÄVINNYT
15 WRITE (NTYP,91) LAST,NAME
GO TO 4
C----- KONE ON VOITTANUT
16 WRITE (NTYP,92) NAME, LAST
GO TO 4
C----- TUTKITAAN, OLLAANKO JOSSAKIN TILANTEISTA 1 - 13, JOS EI, NIIN
C----- TEHDÄÄN SIIRTO CDOTUSLISTAN MUKAAN
12 IF (M1) 12,12,13
DO 14 I A KI,20
M1 A WAIT(I)
IF (MOVE (M1)) 13,13,14
14 CONTINUE
C----- JOS ODOTUSLISTAN KAIKKI RUUDUT OVAT JO VARATTUINA, ON PELI
C----- TASAPELI
19 WRITE (NTYP,96) NAME
GO TO 4
C----- ON LÖYTYNYP VAPAA RUUTU. VARATAAN RUUTU 5-LLÄ
13 MOVE (M1) A 5
CALL BOARD (M1,M1)
CALL BOARD (M2,M2)
C----- TULOSTETAAN KONEEN TEKEMÄ SIIRTO
WRITE (NTYP,99) M2,M2,M3
GOTO 2
END

```

FEATURES SUPPORTED
ONE WORD INTEGERS
IOCS

CORE REQUIREMENTS FOR TTT3D
COMMON 510 VARIABLES 30 PROGRAM 664

END OF COMPILATION

// # LOPPU

22	43	64	1	23	42	61	4	26	39	52	13	27	20	40	16	22	42	62	62	2	23	43	63	3
23	38	53	8	27	42	57	12	26	39	50	15	27	19	31	15	22	39	56	5	28	43	63	2	
22	38	54	6	23	39	55	7	26	42	58	10	27	43	50	11	22	33	24	21	26	27	28	25	
22	26	30	18	23	27	31	19	22	27	34	17	23	26	29	20	38	39	40	37	32	43	44	41	
38	42	45	34	30	43	47	35	38	43	48	33	39	42	45	36	61	1	21	41	64	4	24	44	
49	4	19	34	61	16	31	46	49	13	25	37	52	16	23	40	52	1	10	25	64	13	20	47	
49	1	17	33	52	4	20	36	61	13	29	45	64	16	22	43	4	1	2	3	16	13	14	15	
13	1	5	9	16	4	8	12	16	1	6	11	13	4	7	10	22	49	50	21	64	61	62	63	
61	49	53	57	64	52	56	60	64	49	54	59	61	58	55	58	10	34	59	2	10	35	51	3	
21	37	53	5	24	40	56	8	25	41	57	9	20	44	60	12	30	46	62	14	51	47	63	15	
6	7	8	5	10	11	12	9	6	10	14	2	7	14	15	3	19	19	20	17	20	31	32	29	
21	25	29	17	24	28	32	20	34	35	36	33	46	47	48	45	37	41	45	33	40	44	48	36	
54	55	58	53	58	59	60	57	54	58	52	50	55	52	53	51	22	43	23	42	25	39	27	38	
1	64	13	58	4	61	16	49	22	43	23	42	4	-1	-1	15	-1	1	3	-1	-1	10	10	-1	
10	5	10	2	2	-1	2	1	2	2	1	1	2	0	2	5	9	10	5	5	5	5	0	10	
5	0	5	-1	-1	-1																			

KY

TÄMÄ OHJELMA PELAA KÄSIEN VÄLILLÄ SHAKKIA KOMPIJUTOINTELLISEN LA PELIAUDALLA, JOKA KOOSTUU NUKKESKÄ KRI TASOISSA OLEVISTA 4X4-RUUDUKOISTA. NIKÄLTÄ HALUAT SELOSTUKSEN, MITEN PELIÄ PELATAAN, KIRJOITA KYLLÄ. JOS SAAS KUNNIT PELIN, KIRJOITA EI, JOLLOIN SELOSTUSTA EI OLE. PELEH TARKOITUKSENA ON SAADA 4 CHIA NUKKESKÄ TIEHEN RIVIN. RIVI VOI OLLA VAAKA- TAI PISTYVAARISSA KAIKIVISTÄ RIVIT. SIIRTO TEHDÄÄN ANTAMALLA KODILLE KODI-NUMEROINEN LUKU. ESIMERKIKSI 243 MERKITYS, ETTÄ SIIRTO TEHTÄÄN 2. TASON 4. RIVIN 3. SÄNKIEN SÄNKI. NI-SÄNKIEN SIIRTOVUORO VOIDAAN ANNAA KODIN 3 KIRJOITTAMALLA 000. UUSIEN PELEH PÄIKKÄÄN SIIRTESSÄ -11. PELEMINEN VOIDAAN LOPETTAA TEKEMÄLLÄ SIIRTO 999.

Liite 4. Esimerkki pelatuista pelistä

TÄMÄ OHJELMA PELAA ERMÄISEN SHAKKIA KOLMILOTTISEL-
LA PELILAUDALLA, JOKA KOOSTUU HELJUSSA ERI TASOSSA
OLEVISTA 4x4-RUUDUKOISTA. MIKKLI HALUAT SELOSTUKSEN,
MITEN PELI PELATAAN, KIRJOITA KYLLÄ, JOS TAAS TUHNET
PELIIN, KIRJOITA EI, JOLLOIN SELOSTUSTA EI TULE.
KYLÄ

PELIIN TARKOITUKSENA ON SAADA 4 OMAA MERKKIÄ YHTEEN
RIVIIN. RIVI VOI OLLA VAAKA- TAI PYSTYSUORASSA TAI
LIVISTÄJRIVI, SIIRTO TENDÄKÄ ANTAMALLA KONEELLE KOL-
MIKUMEROINEN LUKU. ESIMERKIKSI 263 MERKITSEE, ETÄ
SIIRTO TEHTIIN 2. TASON 6. RIVIN 3. SARAKKEESEEN. EH-
SIMMÄINEN SIIRTOVUORO VOIDAAN ANTAA KONEELLE KIRJOIT-
TAMALLA 000. URTEEN PELIIN PÄRSTÄKÄ SIIRROLLA -11.
PELAAMISEN VOIDAAN LOPETTAA TEKEMÄLLÄ SIIRTO 099.

ILMOITATKO NIMESI, OLE HYVÄ
MATTI MAANRAKONEN

SINUN VUOROSI, MATTI MAANRAKONEN

111
SIIRTOSI OLI 111
SIIRTONI ON 222 STRATEGIANA 0 14

SINUN VUOROSI, MATTI MAANRAKONEN

115
SIIRTOSI OLI 115
SIIRTONI ON 332 STRATEGIANA 223 13

SINUN VUOROSI, MATTI MAANRAKONEN

442
SIIRTOSI OLI 442
SIIRTONI ON 223 STRATEGIANA 223 13

SINUN VUOROSI, MATTI MAANRAKONEN

224
SIIRTOSI OLI 224
SIIRTONI ON 114 STRATEGIANA 444 3

SINUN VUOROSI, MATTI MAANRAKONEN

114
SIIRTOSI OLI 114
KATSOHAN TÄRKEÄÄ, MATTI MAANRAKONEN, SIIRTOSI ON LAITON

SINUN VUOROSI, MATTI MAANRAKONEN

112
SIIRTOSI OLI 112
VOITIN SINUT MATTI MAANRAKONEN RIVILLÄ 223 332 441 114