

CHURCH-TURING THESIS: THE TURING IMMORTALITY PROBLEM
SOLVED WITH A DYNAMIC REGISTER MACHINE

This section presents a complete formal description of a *dynamic register machine program*, called the IDRM, that determines for any arbitrary Turing Machine, whether it is a halting machine or contains an immortal configuration. The program has 1590 instructions composed from the *Constant* (C m n), *Successor* (S m), *Transfer* (T m n), *Address* (T m n), *Jump* (J m n q), *Delete* (D m n), and *Update* (U m n q) *Instructions*. For the mathematical theory, see sections 1 through 8. For the definition of the dynamic register machine instructions and program execution, see section 9. For the design of this program, see section 10.

<u>Instruction Number</u>	<u>Instruction</u>	<u>Program Comments</u>
0	(A 1 0)	Register 0 stores the register number of the terminating zero register
1	(C 2 0)	that indicates the end of the Turing program.
2	(C 21 6)	Register 21 stores the jump address which is 6
3	(J 1 2 21)	Check if terminating zero register contains zero
4	(C 21 1570)	If not, exit TDRM program with ERROR TURING PROGRAM
5	(J 0 0 21)	
6	(T 9 0)	Terminating zero register is ok. Now check that this register number
7	(C 33 156)	is > 156 where stored Turing program begins.
8	(C 1 0)	
9	(C 21 1570)	
10	(J 1 9 21)	if terminating register <= 156, then exit ERROR TURING PROGRAM
11	(C 21 16)	
12	(J 1 33 21)	
13	(S 1)	
14	(C 21 9)	Register 21 stores the jump address
15	(J 0 0 21)	
16	(C 26 1)	Prepare to find the maximum alphabet and state values in Turing program
17	(C 30 0)	Register 30 stores the maximum alphabet value
18	(C 29 0)	Register 29 stores the maximum state value
19	(C 31 0)	Registers 31 and 32 are currently not used.
20	(C 32 0)	
21	(A 34 33)	Register 34 contains state q in Turing command (q a r b m)
22	(S 33)	
23	(A 35 33)	Register 35 contains alphabet symbol a in Turing command (q a r b m)
24	(S 33)	
25	(A 36 33)	Register 36 contains state r in Turing command (q a r b m)

26	(S 33)	
27	(A 37 33)	Register 37 contains alphabet symbol b in Turing command (q a r b m)
28	(S 33)	
29	(A 38 33)	Register 38 contains move symbol m in Turing command (q a r b m)
30	((C 0 0)	
31	((C 21 77)	If state q = 0, then end of valid Turing program or invalid program.
32	(J 34 0 21)	
33	(T 0 34)	Store new state value from Turing program in register 0
34	(T 1 29)	Register 29 contains the current maximum state value.
35	((C 20 38)	Register 20 stores the continue address 38 after max subroutine exit.
36	((C 21 109)	
37	(J 0 0 21)	Jump to program instruction 109 where the max subroutine starts.
38	(T 29 2)	Copy new maximum state value from register 2 into register 29.
39	((C 0 0)	
40	((C 21 1570)	
41	(J 35 0 21)	If alphabet symbol a = 0 in (q a r b m). ERROR: Invalid Turing program
42	(T 0 35)	Store new alphabet a value in (q a r b m) from Turing program in register 0
43	(T 1 30)	Store current maximum alphabet value in register 1.
44	((C 20 47)	Register 20 stores the continue address 47 after max subroutine exit.
45	((C 21 109)	
46	(J 0 0 21)	Jump to program instruction 109 where the max subroutine starts.
47	(T 30 2)	Copy new maximum alphabet value from register 2 into register 30.
48	((C 0 0)	
49	((C 21 1570)	
50	(J 36 0 21)	If state value r = 0, (q a r b m), exit with ERROR. Invalid Turing program.
51	(T 0 36)	Store current state r from Turing command in register 0
52	(T 1 29)	Store maximum state in register 0
53	((C 20 56)	Register 20 stores the continue address 56 after max subroutine exit.
54	((C 21 109)	
55	(J 0 0 21)	
56	(T 29 2)	Copy new maximum state value from register 2 into register 29.
57	((C 0 0)	
58	((C 21 1570)	
59	(J 37 0 21)	If alphabet symbol b = 0, in (q a r b m) exit ERROR. Invalid Turing program
60	(T 0 37)	Store new alphabet "b" value in (q a r b m) from Turing program in register 0
61	(T 1 30)	Store current maximum alphabet value in register 1.
62	((C 20 65)	
63	((C 21 109)	
64	(J 0 0 21)	Jump to program instruction 109 where the max subroutine starts.
65	(T 30 2)	Copy new maximum alphabet value from register 2 into register 30.
66	((C 0 1)	
67	((C 21 74)	
68	(J 38 0 21)	Check if move symbol m = 1, LEFT move, in (q a r b m)
69	((C 0 2)	
70	((C 21 74)	
71	(J 38 0 21)	Check if move symbol m = 2, RIGHT move, in (q a r b m)
72	((C 21 1570)	In valid move symbol, exit with ERROR. Invalid Turing program
73	(J 0 0 21)	
74	(S 33)	Increment reg 33 to indirectly point to q in next (q a r b m) command.

75	(C 21 21)	
76	(J 0 0 21)	Jump to Line 21 to start checking the next command (q a r b m)
77	(C 33 156)	Turing program commands start at register 156
78	(C 23 1)	Now that maximum state and alphabet values are determined. Use these
79	(T 8 29)	in a state and alphabet loop to check that there is a valid Turing program
80	(S 8)	starting at register 156.
81	(T 9 30)	
82	(S 9)	
83	(S 23)	Start of state Q_COUNT loop. Q_COUNT from 2 to MAX_Q.
84	(C 21 105)	Halt state is 1, so we start Q_COUNT at 2
85	(J 23 8 21)	
86	(C 25 0)	
87	(S 25)	Start of alphabet A_COUNT loop. a from 1 to A = MAX_A
88	(C 21 83)	(q 1 b_q1 r_q1 m_q1), (q 2 b_q2 r_q1 m_q1)
89	(J 25 9 21)	. . . , (q A b_q A r_q A m_q A)
90	(A 34 33)	Store q from (q a r b m) command in register 34
91	(S 33)	
92	(A 35 33)	Store a from (q a r b m) register 35
93	(S 33)	
94	(S 33)	
95	(S 33)	
96	(S 33)	
97	(C 21 101)	
98	(J 34 23 21)	Check that q = Q_COUNT in register 23.
99	(C 21 1570)	
100	(J 0 0 21)	If (q != Q_COUNT), invalid Turing program. Exit with ERROR.
101	(C 21 87)	
102	(J 35 25 21)	Check that a = A_COUNT in register 25.
103	(C 21 1570)	
104	(J 0 0 21)	If (a != A_COUNT), invalid Turing program. Exit with ERROR.
105	(C 18 1)	Register 18 stores the number of insert instructions for Update (U m 18 q)
106	(C 19 1)	Register 19 stores the number of instructions to delete in (D m 19)
107	(C 21 127)	Line 127 is where the A_offsets program is located
108	(J 9 9 21)	Jump to A_offsets
109	(C 21 125)	MAXIMUM PROGRAM. Compares registers 0 and 1.
110	(J 0 1 21)	Compares registers 0 and 1. Returns the maximum of registers
111	(C 2 0)	0 and 1 in register 2.
112	(J 1 2 21)	
113	(C 21 122)	
114	(J 0 2 21)	
115	(S 2)	
116	(C 21 122)	
117	(J 0 2 21)	
118	(C 21 125)	
119	(J 1 2 21)	
120	(C 21 115)	
121	(J 0 0 21)	

122 (T 2 1)
 123 (C 21 126)
 124 (J 0 0 21)
 125 (T 2 0)
 126 (J 0 0 20)

The A_offsets are determined and stored by instructions 127 to 153
 Store A_2, A_3, . . . , A_|Q| at (R (R 39)), (R (+ (R 39) 1)),
 . . . , (R (+ (R 39) |Q| -2))

127 (T 39 33)
 128 (C 0 156)
 129 (T 1 30)
 130 (C 23 1)
 131 (C 5 2)
 132 (T 6 33)
 133 (C 7 0)
 134 (C 22 136)
 135 (U 5 18 22)
 136 (D 22 19)
 137 (C 2 0)
 138 (S 0)
 139 (S 0)
 140 (S 0)
 141 (S 0)
 142 (S 0)
 143 (S 2)
 144 (C 21 148)
 145 (J 1 2 21)
 146 (C 21 138)
 147 (J 1 1 21)
 148 (S 33)
 149 (S 23)
 150 (C 21 154)
 151 (J 23 29 21)
 152 (C 21 131)
 153 (J 0 0 21)

Register 156 contains 2 which is the first Turing command (2 1 r b m)

Store A_k in register

Add 5 to Register 0 so that it points to the next Turing Command

Instructions 154 to 296 set up a scratch pad of registers to execute prime directed edge search method 8.34 described in section 8.

Instructions 154 to 167 set up register to record prime input command sequence.

154 (S 33)
 155 (T 48 33)
 156 (S 33)
 157 (T 50 33)
 158 (C 0 0)
 159 (T 1 29)
 160 (S 1)
 161 (S 0)
 162 (S 33)
 163 (S 33)
 164 (C 21 168)
 165 (J 0 1 21)

166	(C 21 161)	
167	(J 0 0 21)	
168	(S 33)	Instructions 168 to 190 set up registers to record current prospective head node.
169	(T 56 33)	
170	(C 0 1)	
171	(C 28 0)	
172	(T 1 29)	
173	(S 0)	
174	(S 28)	
175	(S 33)	
176	(C 21 180)	
177	(J 0 1 21)	
178	(C 21 173)	
179	(J 0 0 21)	
180	(T 57 33)	
181	(C 0 1)	
182	(T 1 29)	
183	(S 0)	
184	(S 28)	
185	(S 33)	
186	(C 21 190)	
187	(J 0 1 21)	
188	(C 21 183)	
189	(J 0 0 21)	
190	(T 58 33)	
191	(S 28)	Instruction 191 to 212 set up registers to record the current prospective tail node.
192	(S 33)	
193	(S 33)	
194	(T 62 33)	
195	(C 0 1)	
196	(T 1 29)	
197	(S 0)	
198	(S 33)	
199	(C 21 203)	
200	(J 0 1 21)	
201	(C 21 197)	
202	(J 0 0 21)	
203	(T 63 33)	
204	(C 0 1)	
205	(T 1 29)	
206	(S 0)	
207	(S 33)	
208	(C 21 212)	
209	(J 0 1 21)	
210	(C 21 206)	
211	(J 0 0 21)	
212	(T 64 33)	
213	(S 33)	Instructions 213 to 233 set up registers to record the execution tape during the computation of a prospective prime directed edge.
214	(S 33)	

215 (T 68 33)
 216 (C 0 1)
 217 (T 1 29)
 218 (S 0)
 219 (S 33)
 220 (C 21 224)
 221 (J 0 1 21)
 222 (C 21 218)
 223 (J 0 0 21)
 224 (T 69 33)
 225 (C 0 1)
 226 (T 1 29)
 227 (S 0)
 228 (S 33)
 229 (C 21 233)
 230 (J 0 1 21)
 231 (C 21 227)
 232 (J 0 0 21)
 233 (T 70 33)
 234 (S 33)
 235 (S 33)
 236 (T 72 33)
 237 (C 0 1)
 238 (T 1 29)
 239 (S 0)
 240 (S 33)
 241 (C 21 245)
 242 (J 0 1 21)
 243 (C 21 239)
 244 (J 0 0 21)
 245 (T 73 33)
 246 (C 0 1)
 247 (T 1 29)
 248 (S 0)
 249 (S 33)
 250 (C 21 254)
 251 (J 0 1 21)
 252 (C 21 248)
 253 (J 0 0 21)
 254 (T 74 33)
 255 (S 33)
 256 (S 33)
 257 (T 46 33)
 258 (C 0 0)
 259 (T 1 29)
 260 (S 0)
 261 (S 33)
 262 (C 21 266)
 263 (J 0 1 21)

Instructions 234 to 254 set up registers to store an iteration tape, which is used to iterate through every tape pattern in $A^{(2|Q|+1)}$

Instructions 255 to 265 set up registers to record the tape head moves during computation of the prospective prime edge.

264 (C 21 260)
 265 (J 0 0 21)
 266 (S 33)
 267 (S 33)
 268 (T 43 33)
 269 (C 0 0)
 270 (T 1 28)
 271 (C 2 1)
 272 (C 5 2)
 273 (C 7 0)
 274 (T 6 33)
 275 (C 22 277)
 276 (U 5 18 22)
 277 (D 22 19)
 278 (S 0)
 279 (S 2)
 280 (S 33)
 281 (C 21 290)
 282 (J 0 1 21)
 283 (C 21 287)
 284 (J 2 29 21)
 285 (C 21 274)
 286 (J 0 0 21)
 287 (T 44 33)
 288 (C 21 274)
 289 (J 0 0 21)
 290 (S 33)
 291 (S 33)
 292 (A 27 44)
 293 (C 81 0)
 294 (T 83 33)
 295 (T 85 33)
 296 (T 155 33)

 297 (C 71 1)
 298 (C 3 1)
 299 (T 2 72)
 300 (C 0 0)
 301 (T 1 28)
 302 (C 5 2)
 303 (C 7 3)
 304 (T 6 2)
 305 (C 22 307)
 306 (U 5 18 22)
 307 (D 22 19)
 308 (S 0)
 309 (S 2)
 310 (C 21 314)

Instructions 266 to 292 set up registers to compute the window of execution.

Instructions 293 to 296 set up memory pointers to record the number of prime edges found so far and store a linked list of prime edges.

In Instruction 296 a free heap pointer is stored in register 155.

Instructions 297 to 858 perform the Prime directed edge search method as defined in 8.34

Instructions 297 to 313 initialize the tape $A^2|Q|+1$ to all 1's.

311 (J 0 1 21)
 312 (C 21 304)
 313 (J 0 0 21)
 314 (C 0 1)
 315 (T 33 72)
 316 (C 1 0)
 317 (C 21 328)
 318 (J 1 28 21)
 319 (A 3 33)
 320 (C 21 324)
 321 (J 0 3 21)
 322 (C 21 331)
 323 (J 0 0 21)
 324 (S 1)
 325 (S 33)
 326 (C 21 317)
 327 (J 0 0 21)
 328 (C 21 858)
 329 (J 71 29 21)
 330 (S 71)
 331 (T 65 71)
 332 (T 0 68)
 333 (T 1 72)
 334 (T 2 28)
 335 (C 3 0)
 336 (C 5 2)
 337 (C 21 349)
 338 (J 2 3 21)
 339 (T 6 0)
 340 (T 7 1)
 341 (C 22 343)
 342 (U 5 18 22)
 343 (D 22 19)
 344 (S 0)
 345 (S 1)
 346 (S 3)
 347 (C 21 337)
 348 (J 0 0 21)
 349 (A 66 73)
 350 (T 53 65)
 351 (T 0 56)
 352 (T 1 72)
 353 (T 2 28)
 354 (C 3 0)
 355 (C 5 2)
 356 (C 21 368)
 357 (J 2 3 21)
 358 (T 6 0)
 359 (T 7 1)

Instructions 331 to 349 copy the iterated tape pattern, starting at the register pointed to by register 68.

Instruction 334 stores the value of $2|Q|+1$ in register 2.

Instructions 350 to 367 initialize the prospective head node state and the tape symbols.

360 (C 22 362)
 361 (U 5 18 22)
 362 (D 22 19)
 363 (S 0)
 364 (S 1)
 365 (S 3)
 366 (C 21 356)
 367 (J 0 0 21)
 368 (C 48 0)
 369 (T 67 69)
 370 (T 51 50)
 371 (C 21 551)
 372 (J 65 26 21)
 373 (C 0 2)
 374 (T 33 39)
 375 (C 21 381)
 376 (J 65 0 21)
 377 (S 0)
 378 (S 33)
 379 (C 21 375)
 380 (J 0 0 21)
 381 (A 0 33)
 382 (T 33 0)
 383 (T 1 33)
 384 (S 1)
 385 (A 35 1)
 386 (C 21 395)
 387 (J 35 66 21)
 388 (S 33)
 389 (S 33)
 390 (S 33)
 391 (S 33)
 392 (S 33)
 393 (C 21 383)
 394 (J 0 0 21)
 395 (T 47 46)
 396 (C 5 2)
 397 (T 6 51)
 398 (C 7 65)
 399 (C 22 401)
 400 (U 5 18 22)
 401 (D 22 19)
 402 (S 51)
 403 (T 6 51)
 404 (T 7 67)
 405 (C 22 407)
 406 (U 5 18 22)
 407 (D 22 19)
 408 (S 51)

Instructions 371 to 395 initialize register 33 to the correct command (q a r b m) in the Turing command table. The current state q of the Turing machine is stored in register 65. The current tape symbol a is stored in register 66.

Instructions 396 to 408 store (q_k, a_k) in the register that is pointed to by the contents of register 51.

409 (A 34 33)
410 (S 33)
411 (A 35 33)
412 (S 33)
413 (A 36 33)
414 (S 33)
415 (A 37 33)
416 (S 33)
417 (A 38 33)
418 (A 66 67)
419 (C 21 423)
420 (J 66 35 21)
421 (C 21 1570)
422 (J 0 0 21)
423 (T 65 36)
424 (T 66 37)
425 (C 5 2)
426 (T 6 67)
427 (C 7 37)
428 (C 22 430)
429 (U 5 18 22)
430 (D 22 19)
431 (C 21 551)
432 (J 65 26 21)
433 (C 5 2)
434 (T 6 47)
435 (C 7 4)
436 (C 0 1)
437 (C 21 446)
438 (J 38 0 21)
439 (S 67)
440 (C 4 2)
441 (C 22 443)
442 (U 5 18 22)
443 (D 22 19)
444 (C 21 460)
445 (J 0 0 21)
446 (C 4 1)
447 (C 22 449)
448 (U 5 18 22)
449 (D 22 19)
450 (T 0 68)
451 (T 1 0)
452 (S 1)
453 (C 21 459)
454 (J 1 67 21)
455 (S 1)
456 (S 0)
457 (C 21 453)

Instructions 409 to 483 execute one computational step of the Turing machine whose command table starts at register 156.

458 (J 0 0 21)
 459 (T 67 0)
 460 (S 47)
 461 (A 66 67)
 462 (C 0 2)
 463 (T 33 39)
 464 (C 21 470)
 465 (J 65 0 21)
 466 (S 0)
 467 (S 33)
 468 (C 21 464)
 469 (J 0 0 21)
 470 (A 0 33)
 471 (T 33 0)
 472 (T 1 33)
 473 (S 1)
 474 (A 35 1)
 475 (C 21 484)
 476 (J 66 35 21)
 477 (S 33)
 478 (S 33)
 479 (S 33)
 480 (S 33)
 481 (S 33)
 482 (C 21 472)
 483 (J 0 0 21)
 484 (S 48)
 485 (C 1 1)
 486 (T 52 50)
 487 (A 0 52)
 488 (C 21 497)
 489 (J 0 65 21)
 490 (C 21 396)
 491 (J 1 48 21)
 492 (S 52)
 493 (S 52)
 494 (S 1)
 495 (C 21 487)
 496 (J 0 0 21)
 497 (C 5 2)
 498 (T 6 51)
 499 (C 7 65)
 500 (C 22 502)
 501 (U 5 18 22)
 502 (D 22 19)
 503 (S 51)
 504 (T 6 51)
 505 (C 7 66)
 506 (C 22 508)

Instruction 484 increments register 48 which stores the number of computational steps for this prospective prime directed edge.

Instructions 485 to 496 check to see if state $q_{(k+1)}$ has already been visited.

If state $q_{(k+1)}$ has already been visited, jump to 497

Instructions 497 to 509 store (q_N, a_N) starting at the register pointed to by the contents of register 51.

507 (U 5 18 22)
 508 (D 22 19)
 509 (S 51)
 510 (T 47 46)
 511 (T 40 44)
 512 (T 41 44)
 513 (T 45 44)
 514 (C 1 1)
 515 (C 2 2)
 516 (C 3 0)
 517 (A 0 47)
 518 (C 21 529)
 519 (J 0 1 21)
 520 (C 21 525)
 521 (J 41 45 21)
 522 (S 45)
 523 (C 21 545)
 524 (J 0 0 21)
 525 (S 45)
 526 (S 41)
 527 (C 21 545)
 528 (J 0 0 21)
 529 (T 9 45)
 530 (C 4 1)
 531 (A 5 45)
 532 (T 6 43)
 533 (C 21 539)
 534 (J 4 5 21)
 535 (S 4)
 536 (S 6)
 537 (C 21 533)
 538 (J 0 0 21)
 539 (T 45 6)
 540 (C 21 544)
 541 (J 40 9 21)
 542 (C 21 545)
 543 (J 0 0 21)
 544 (T 40 45)
 545 (S 3)
 546 (S 47)
 547 (C 21 551)
 548 (J 48 3 21)
 549 (C 21 517)
 550 (J 0 0 21)
 551 (T 33 72)
 552 (C 0 0)
 553 (T 1 28)
 554 (A 2 33)
 555 (C 21 579)

Instructions 510 to 550 compute the window of execution.

Instructions 551 to 578 increment the tape pattern which is an element of $A^{(2|Q|+1)}$. This enables the program to search every element in $A^{(2|Q|+1)}$ as a prospective head node.

556 (J 0 1 21)
557 (C 21 568)
558 (J 2 30 21)
559 (S 2)
560 (C 5 2)
561 (T 6 33)
562 (C 7 2)
563 (C 22 565)
564 (U 5 18 22)
565 (D 22 19)
566 (C 21 579)
567 (J 0 0 21)
568 (C 5 2)
569 (T 6 33)
570 (C 2 1)
571 (C 7 2)
572 (C 22 574)
573 (U 5 18 22)
574 (D 22 19)
575 (S 0)
576 (S 33)
577 (C 21 554)
578 (J 0 0 21)
579 (C 21 856)
580 (J 65 26 21)
581 (T 82 81)
582 (S 82)
583 (T 33 85)
584 (S 33)
585 (C 5 2)
586 (T 6 33)
587 (C 7 82)
588 (C 22 590)
589 (U 5 18 22)
590 (D 22 19)
591 (S 33)
592 (T 88 33)
593 (S 33)
594 (T 89 33)
595 (S 33)
596 (T 90 33)
597 (S 33)
598 (C 42 0)
599 (C 54 0)
600 (C 60 0)
601 (T 0 40)
602 (C 21 606)
603 (J 0 44 21)
604 (C 21 607)

Instructions 579 to 778 copy this new prime directed edge found
to the end of the prime directed edge linked list.

605 (J 0 0 21)
606 (T 54 42)
607 ((C 21 611)
608 (J 0 45 21)
609 ((C 21 612)
610 (J 0 0 21)
611 (T 60 42)
612 ((C 21 618)
613 (J 0 41 21)
614 (S 42)
615 (S 0)
616 ((C 21 602)
617 (J 0 0 21)
618 ((C 5 2)
619 (T 6 33)
620 ((C 7 42)
621 ((C 22 623)
622 (U 5 18 22)
623 (D 22 19)
624 (S 33)
625 ((C 5 2)
626 (T 6 33)
627 ((C 7 71)
628 ((C 22 630)
629 (U 5 18 22)
630 (D 22 19)
631 ((C 5 2)
632 (T 6 88)
633 ((C 7 33)
634 ((C 22 636)
635 (U 5 18 22)
636 (D 22 19)
637 (S 33)
638 ((C 5 2)
639 (T 6 33)
640 ((C 7 54)
641 ((C 22 643)
642 (U 5 18 22)
643 (D 22 19)
644 (S 33)
645 (T 55 33)
646 (S 33)
647 (T 77 33)
648 (T 78 56)
649 ((C 0 0)
650 (A 1 40)
651 ((C 21 657)
652 (J 0 1 21)
653 (S 0)

654 (S 78)
655 (C 21 651)
656 (J 0 0 21)
657 (T 2 42)
658 (S 2)
659 (C 1 0)
660 (C 5 2)
661 (C 21 676)
662 (J 54 1 21)
663 (C 21 683)
664 (J 2 1 21)
665 (T 6 77)
666 (T 7 78)
667 (C 22 669)
668 (U 5 18 22)
669 (D 22 19)
670 (S 77)
671 (S 78)
672 (S 33)
673 (S 1)
674 (C 21 661)
675 (J 0 0 21)
676 (T 6 55)
677 (C 7 77)
678 (C 22 680)
679 (U 5 18 22)
680 (D 22 19)
681 (C 21 663)
682 (J 0 0 21)
683 (C 5 2)
684 (T 6 33)
685 (C 7 65)
686 (C 22 688)
687 (U 5 18 22)
688 (D 22 19)
689 (C 5 2)
690 (T 6 89)
691 (C 7 33)
692 (C 22 694)
693 (U 5 18 22)
694 (D 22 19)
695 (S 33)
696 (C 5 2)
697 (T 6 33)
698 (C 7 60)
699 (C 22 701)
700 (U 5 18 22)
701 (D 22 19)
702 (S 33)

703 (T 61 33)
704 (S 33)
705 (T 77 33)
706 (T 78 68)
707 (C 0 0)
708 (A 1 40)
709 (C 21 715)
710 (J 0 1 21)
711 (S 0)
712 (S 78)
713 (C 21 709)
714 (J 0 0 21)
715 (T 2 42)
716 (S 2)
717 (C 1 0)
718 (C 5 2)
719 (C 21 734)
720 (J 60 1 21)
721 (C 21 741)
722 (J 2 1 21)
723 (T 6 77)
724 (T 7 78)
725 (C 22 727)
726 (U 5 18 22)
727 (D 22 19)
728 (S 77)
729 (S 78)
730 (S 33)
731 (S 1)
732 (C 21 719)
733 (J 0 0 21)
734 (T 6 61)
735 (C 7 77)
736 (C 22 738)
737 (U 5 18 22)
738 (D 22 19)
739 (C 21 721)
740 (J 0 0 21)
741 (C 0 0)
742 (T 49 48)
743 (C 21 749)
744 (J 48 0 21)
745 (S 0)
746 (S 49)
747 (C 21 743)
748 (J 0 0 21)
749 (C 5 2)
750 (T 6 33)
751 (C 7 49)

752 (C 22 754)
 753 (U 5 18 22)
 754 (D 22 19)
 755 (C 5 2)
 756 (T 6 90)
 757 (C 7 33)
 758 (C 22 760)
 759 (U 5 18 22)
 760 (D 22 19)
 761 (S 33)
 762 (T 77 33)
 763 (T 78 50)
 764 (C 0 0)
 765 (S 49)
 766 (T 6 77)
 767 (T 7 78)
 768 (C 22 770)
 769 (U 5 18 22)
 770 (D 22 19)
 771 (C 21 779)
 772 (J 49 0 21)
 773 (S 77)
 774 (S 78)
 775 (S 33)
 776 (S 0)
 777 (C 21 766)
 778 (J 0 0 21)
 779 (S 33)
 780 (T 84 83)
 781 (C 21 840)
 782 (J 84 85 21)
 783 (T 77 84)
 784 (T 78 85)
 785 (S 77)
 786 (S 77)
 787 (S 77)
 788 (S 77)
 789 (S 77)
 790 (S 78)
 791 (S 78)
 792 (S 78)
 793 (S 78)
 794 (S 78)
 795 (A 1 77)
 796 (A 2 78)
 797 (C 21 801)
 798 (J 1 2 21)
 799 (C 21 836)
 800 (J 0 0 21)

Instructions 779 and 780 update the prime directed edge linked list pointers. Instructions 781 to 839 check if the new head node just found is the same as the head node at the end of the prime directed edge linked list. If yes, then the new prime directed edges is the same as the last one found and it is ignored.

801 (S 77)
802 (S 78)
803 (A 75 77)
804 (A 76 78)
805 (C 21 809)
806 (J 75 76 21)
807 (C 21 836)
808 (J 0 0 21)
809 (S 77)
810 (S 78)
811 (A 79 77)
812 (A 80 78)
813 (C 21 817)
814 (J 79 80 21)
815 (C 21 836)
816 (J 0 0 21)
817 (S 77)
818 (S 78)
819 (S 77)
820 (S 78)
821 (C 0 0)
822 (S 1)
823 (A 79 77)
824 (A 80 78)
825 (C 21 829)
826 (J 79 80 21)
827 (C 21 836)
828 (J 0 0 21)
829 (S 77)
830 (S 78)
831 (S 0)
832 (C 21 856)
833 (J 0 1 21)
834 (C 21 823)
835 (J 0 0 21)
836 (A 0 84)
837 (T 84 0)
838 (C 21 781)
839 (J 0 0 21)
840 (C 5 2)
841 (T 6 85)
842 (C 7 33)
843 (C 22 845)
844 (U 5 18 22)
845 (D 22 19)
846 (T 85 33)
847 (S 33)
848 (T 155 33)
849 (C 0 0)

The prime directed edge just found is new and so it is added to the end of the primed directed edge linked list. This occurs in instructions 840 to 854.

850	(T 6 85)	
851	(C 7 0)	
852	(C 22 854)	
853	(U 5 18 22)	
854	(D 22 19)	
855	(S 81)	The number of prime directed edges found so far is incremented
856	(C 21 314)	and stored in register 81.
857	(J 0 0 21)	Jump back to instruction 314 where the next tape pattern is examined
		to decide if the current state and new tape pattern determine a new
		prime directed edge.
858	(T 86 155)	Instructions 858 to 875 set up an array of m prime edge pointers.
859	(T 33 86)	
860	(T 84 83)	
861	(C 0 0)	
862	(C 5 2)	
863	(C 21 876)	
864	(J 0 81 21)	
865	(C 7 84)	
866	(T 6 33)	
867	(C 22 869)	
868	(U 5 18 22)	
869	(D 22 19)	
870	(S 0)	
871	(S 33)	
872	(A 2 84)	
873	(T 84 2)	
874	(C 21 863)	
875	(J 0 0 21)	
876	(C 0 0)	Instructions 876, 877 test if register 0 contains m, the number of prime
877	(T 155 33)	directed edges.
878	(S 155)	Instructions 878 to 85 initialize registers so that register 33 points to the next
879	(T 33 86)	prime directed edge pointer, register 155 points to the free heap, and
880	(T 91 155)	register 99 points to the head node of the edge sequence linked list.
881	(S 155)	
882	(T 99 155)	
883	(T 93 99)	
884	(C 96 1)	Register 96 stores the number of current edge sequences in E(k)
885	(C 97 1)	Also, register 97 is initialized to 1 and stores the number of distinct prime
886	(C 21 1017)	edges composed together in every element of E(1)
887	(J 0 81 21)	
888	(T 94 93)	Instructions 888 to 903 set up the h_node, t_node and s_node pointers.
889	(S 93)	
890	(A 77 33)	
891	(S 77)	
892	(S 77)	
893	(T 88 77)	
894	(S 77)	
895	(T 89 77)	
896	(S 77)	

897	(T 90 77)	
898	(C 5 2)	
899	(T 6 93)	
900	(A 7 90)	
901	(C 22 903)	
902	(U 5 18 22)	
903	(D 22 19)	
904	(S 93)	Instructions 904 to 910 store state r_j.
905	(T 6 93)	
906	(A 7 89)	
907	(C 22 909)	
908	(U 5 18 22)	
909	(D 22 19)	
910	(S 93)	
911	(T 6 93)	Instructions 911 to 918 store idx_j.
912	(A 78 89)	
913	(S 78)	
914	(T 7 78)	
915	(C 22 917)	
916	(U 5 18 22)	
917	(D 22 19)	
918	(A 102 78)	
919	(S 93)	Instructions 919 to 926 store ub_j.
920	(T 6 93)	
921	(S 77)	
922	(T 7 77)	
923	(C 22 925)	
924	(U 5 18 22)	
925	(D 22 19)	
926	(A 103 77)	
927	(S 93)	Instructions 927 to 938 store lb_ptr_j.
928	(T 6 93)	
929	(T 77 93)	
930	(S 77)	
931	(S 77)	
932	(S 77)	
933	(S 77)	
934	(C 7 77)	
935	(C 22 937)	
936	(U 5 18 22)	
937	(D 22 19)	
938	(T 104 77)	
939	(C 1 0)	Instructions 939 to 945 compute idx_ptr_j.
940	(C 21 946)	
941	(J 1 102 21)	
942	(S 77)	
943	(S 1)	
944	(C 21 940)	
945	(J 0 0 21)	

946	(S 93)	Instructions 946 to 951 store <code>idx_ptr_j</code> .
947	(T 6 93)	
948	((C 7 77)	
949	(C 22 951)	
950	(U 5 18 22)	
951	(D 22 19)	
952	((C 21 958)	Instructions 952 to 957 compute <code>ub_ptr_j</code> .
953	(J 1 103 21)	
954	(S 77)	
955	(S 1)	
956	((C 21 952)	
957	(J 0 0 21)	
958	(S 93)	Instructions 958 to 963 store <code>ub_ptr_j</code> .
959	(T 6 93)	
960	((C 7 77)	
961	(C 22 963)	
962	(U 5 18 22)	
963	(D 22 19)	
964	(S 93)	Instructions 964 to 969 store <code>p_1,j</code> which equals <code>j</code>
965	(T 6 93)	
966	((C 7 96)	
967	(C 22 969)	
968	(U 5 18 22)	
969	(D 22 19)	
970	(S 93)	Instructions 970 to 987 store <code>w_0,j, . . . , w_idx,j, . . . , W_ub,j</code>
971	((C 1 0)	
972	(A 77 89)	
973	(S 77)	
974	(S 77)	
975	(S 77)	
976	(T 6 93)	
977	(T 7 77)	
978	((C 22 980)	
979	(U 5 18 22)	
980	(D 22 19)	
981	((C 21 988)	
982	(J 1 103 21)	
983	(S 1)	
984	(S 93)	
985	(S 77)	
986	((C 21 976)	
987	(J 0 0 21)	
988	(S 93)	Instructions 988 to 1004 store <code>(q0,j, a_0,j) . . . , (q_Nj,j, a_Nj,j)</code>
989	(A 77 90)	
990	((C 1 1)	
991	(A 49 77)	
992	(S 77)	
993	(T 6 93)	
994	(T 7 77)	

995	(C 22 997)	
996	(U 5 18 22)	
997	(D 22 19)	
998	(C 21 1005)	
999	(J 1 49 21)	
1000	(S 1)	
1001	(S 93)	
1002	(S 77)	
1003	(C 21 993)	
1004	(J 0 0 21)	
1005	(S 93)	Instructions 1005 to 1016 update pointers and pointers to jump back to
1006	(T 6 94)	instruction 886
1007	(C 7 93)	
1008	(C 22 1010)	
1009	(U 5 18 22)	
1010	(D 22 19)	
1011	(S 0)	
1012	(S 33)	
1013	(S 96)	
1014	(T 155 93)	
1015	(C 21 886)	Jump back to instruction 886 and repeat this loop for the j+1st prime edge
1016	(J 0 0 21)	that is stored in E(1)
1017	(T 101 81)	
1018	(T 98 101)	Instruction 1018 is the start of the loop where E(k+1) is constructed from E(k)
1019	(C 101 0)	
1020	(C 96 0)	
1021	(C 21 1551)	Check if E(k) = empty set i.e. if the number of edge sequences = 0.
1022	(J 96 98 21)	If E(k) = empty set, jump to instruction 1551. Write HALT in register 0 to 4.
1023	(T 92 99)	
1024	(T 94 92)	
1025	(T 99 155)	
1026	(T 100 99)	
1027	(C 21 1018)	
1028	(J 96 98 21)	Instruction 1028 corresponds to the beginning of the for loop
1029	(C 87 0)	for each E([p_1, p_2, ..., p_k], k) in E(k) in method 8.34
1030	(T 84 86)	
1031	(C 21 1545)	
1032	(J 87 81 21)	Instruction 1032 is the beginning of the for loop
1033	(A 33 84)	for each prime directed edge (H_j ==> T_j) in prime edge set P
1034	(S 87)	in method 8.34
1035	(T 93 94)	
1036	(S 93)	
1037	(A 137 93)	
1038	(S 93)	
1039	(A 105 93)	
1040	(S 93)	
1041	(A 109 93)	
1042	(S 93)	
1043	(A 111 93)	

1044 (S 93)
 1045 (A 113 93)
 1046 (S 93)
 1047 (A 115 93)
 1048 (S 93)
 1049 (A 117 93)
 1050 (C 107 0)
 1051 (S 33)
 1052 (A 138 33)
 1053 (S 33)
 1054 (T 88 33)
 1055 (S 33)
 1056 (T 89 33)
 1057 (S 33)
 1058 (T 90 33)
 1059 (S 33)
 1060 (A 112 33)
 1061 (A 33 88)
 1062 (A 106 33)
 1063 (S 33)
 1064 (A 110 33)
 1065 (S 33)
 1066 (A 116 33)
 1067 (S 33)
 1068 (T 114 33)
 1069 (C 108 0)
 1070 (C 21 1074)
 1071 (J 105 106 21)
 1072 (C 21 1173)
 1073 (J 0 0 21)
 1074 (T 118 115)
 1075 (T 119 116)
 1076 (T 1 109)
 1077 (T 2 110)
 1078 (A 3 118)
 1079 (A 4 119)
 1080 (C 21 1084)
 1081 (J 3 4 21)
 1082 (C 21 1173)
 1083 (J 0 0 21)
 1084 (C 21 1094)
 1085 (J 1 111 21)
 1086 (C 21 1098)
 1087 (J 2 112 21)
 1088 (S 1)
 1089 (S 2)
 1090 (S 118)
 1091 (S 119)
 1092 (C 21 1078)

If $(H_j \implies T_j)$ link matches with $N_p(k)$ in method 8.34 is executed starting at instruction 1070
 Instructions 1070 to 1461 test for an overlap match between the current prime edge found in the loop beginning at instruction 1032 and the exec

1093 (J 0 0 21)
1094 (C 125 1)
1095 (C 126 0)
1096 (C 21 1100)
1097 (J 0 0 21)
1098 (C 125 0)
1099 (C 126 1)
1100 (T 1 107)
1101 (T 2 108)
1102 (C 0 0)
1103 (C 21 1112)
1104 (J 1 109 21)
1105 (C 21 1136)
1106 (J 2 110 21)
1107 (S 0)
1108 (S 1)
1109 (S 2)
1110 (C 21 1103)
1111 (J 0 0 21)
1112 (C 127 1)
1113 (C 128 0)
1114 (T 118 113)
1115 (C 0 0)
1116 (C 21 1122)
1117 (J 107 0 21)
1118 (S 0)
1119 (S 118)
1120 (C 21 1116)
1121 (J 0 0 21)
1122 (T 119 114)
1123 (C 3 0)
1124 (C 21 1130)
1125 (J 108 3 21)
1126 (S 3)
1127 (S 119)
1128 (C 21 1124)
1129 (J 0 0 21)
1130 (C 21 1160)
1131 (J 2 110 21)
1132 (S 119)
1133 (S 2)
1134 (C 21 1130)
1135 (J 0 0 21)
1136 (C 127 0)
1137 (C 128 1)
1138 (T 119 114)
1139 (C 0 0)
1140 (C 21 1146)
1141 (J 108 0 21)

1142 (S 0)
1143 (S 119)
1144 (C 21 1140)
1145 (J 0 0 21)
1146 (T 118 113)
1147 (C 3 0)
1148 (C 21 1154)
1149 (J 107 3 21)
1150 (S 3)
1151 (S 118)
1152 (C 21 1148)
1153 (J 0 0 21)
1154 (C 21 1160)
1155 (J 1 109 21)
1156 (S 118)
1157 (S 1)
1158 (C 21 1154)
1159 (J 0 0 21)
1160 (C 21 1176)
1161 (J 118 115 21)
1162 (J 119 116 21)
1163 (A 3 118)
1164 (A 4 119)
1165 (C 21 1169)
1166 (J 3 4 21)
1167 (C 21 1173)
1168 (J 0 0 21)
1169 (S 118)
1170 (S 119)
1171 (C 21 1160)
1172 (J 0 0 21)
1173 (C 120 0)
1174 (C 21 1542)
1175 (J 0 0 21)
1176 (C 120 1)
1177 (C 121 0)
1178 (C 122 0)
1179 (C 123 0)
1180 (C 124 0)
1181 (C 21 1189)
1182 (J 120 127 21)
1183 (C 21 1195)
1184 (J 120 126 21)
1185 (C 21 1198)
1186 (J 120 125 21)
1187 (C 21 1361)
1188 (J 0 0 21)
1189 (C 21 1201)
1190 (J 120 125 21)

1191 (C 21 1204)
1192 (J 120 126 21)
1193 (C 21 1361)
1194 (J 0 0 21)
1195 (C 121 1)
1196 (C 21 1205)
1197 (J 0 0 21)
1198 (C 122 1)
1199 (C 21 1205)
1200 (J 0 0 21)
1201 (C 123 1)
1202 (C 21 1205)
1203 (J 0 0 21)
1204 (C 124 1)
1205 (T 132 100)
1206 (S 132)
1207 (S 132)
1208 (S 132)
1209 (S 132)
1210 (S 132)
1211 (S 132)
1212 (S 132)
1213 (C 3 0)
1214 (S 132)
1215 (S 3)
1216 (S 132)
1217 (C 21 1221)
1218 (J 3 97 21)
1219 (C 21 1215)
1220 (J 0 0 21)
1221 (S 132)
1222 (C 145 0)
1223 (A 78 89)
1224 (A 129 78)
1225 (S 78)
1226 (A 146 78)
1227 (S 78)
1228 (S 78)
1229 (T 142 78)
1230 (T 147 112)
1231 (C 21 1239)
1232 (J 120 121 21)
1233 (J 120 122 21)
1234 (C 21 1308)
1235 (J 120 123 21)
1236 (J 120 124 21)
1237 (C 21 1542)
1238 (J 0 0 21)
1239 (T 0 110)

1240 (C 135 0)
1241 (C 21 1247)
1242 (J 0 109 21)
1243 (S 0)
1244 (S 135)
1245 (C 21 1241)
1246 (J 0 0 21)
1247 (C 1 0)
1248 (T 118 113)
1249 (T 131 132)
1250 (C 5 2)
1251 (C 139 0)
1252 (C 3 0)
1253 (C 141 0)
1254 (T 6 131)
1255 (T 7 118)
1256 (C 22 1258)
1257 (U 5 18 22)
1258 (D 22 19)
1259 (S 131)
1260 (S 118)
1261 (S 1)
1262 (S 3)
1263 (S 141)
1264 (C 21 1268)
1265 (J 1 135 21)
1266 (C 21 1254)
1267 (J 0 0 21)
1268 (C 1 0)
1269 (T 144 142)
1270 (T 6 131)
1271 (T 7 144)
1272 (C 22 1274)
1273 (U 5 18 22)
1274 (D 22 19)
1275 (C 21 1287)
1276 (J 1 146 21)
1277 (C 21 1291)
1278 (J 1 147 21)
1279 (S 131)
1280 (S 144)
1281 (S 118)
1282 (S 1)
1283 (S 3)
1284 (S 141)
1285 (C 21 1270)
1286 (J 0 0 21)
1287 (T 140 3)
1288 (T 133 131)

1289 (C 21 1277)
1290 (J 0 0 21)
1291 (C 21 1361)
1292 (J 120 122 21)
1293 (S 131)
1294 (S 118)
1295 (T 6 131)
1296 (T 7 118)
1297 (C 22 1299)
1298 (U 5 18 22)
1299 (D 22 19)
1300 (C 21 1361)
1301 (J 3 111 21)
1302 (S 131)
1303 (S 118)
1304 (S 3)
1305 (S 141)
1306 (C 21 1295)
1307 (J 0 0 21)
1308 (C 1 0)
1309 (T 3 109)
1310 (C 5 2)
1311 (C 141 0)
1312 (T 131 132)
1313 (T 144 142)
1314 (T 6 131)
1315 (T 7 144)
1316 (C 22 1318)
1317 (U 5 18 22)
1318 (D 22 19)
1319 (C 21 1334)
1320 (J 1 146 21)
1321 (C 21 1338)
1322 (J 3 110 21)
1323 (C 21 1342)
1324 (J 1 147 21)
1325 (S 131)
1326 (S 144)
1327 (S 118)
1328 (S 1)
1329 (S 3)
1330 (S 4)
1331 (S 141)
1332 (C 21 1314)
1333 (J 0 0 21)
1334 (T 140 146)
1335 (T 133 131)
1336 (C 21 1321)
1337 (J 0 0 21)

1338 (T 118 113)
1339 (C 4 0)
1340 (C 21 1323)
1341 (J 0 0 21)
1342 (C 21 1361)
1343 (J 120 123 21)
1344 (S 131)
1345 (S 118)
1346 (T 6 131)
1347 (T 7 118)
1348 (C 22 1350)
1349 (U 5 18 22)
1350 (D 22 19)
1351 (C 21 1361)
1352 (J 4 111 21)
1353 (S 131)
1354 (S 118)
1355 (S 4)
1356 (S 141)
1357 (C 21 1346)
1358 (J 0 0 21)
1359 (C 21 1361)
1360 (J 0 0 21)
1361 (T 130 129)
1362 (T 134 131)
1363 (S 131)
1364 (T 136 131)
1365 (T 77 117)
1366 (S 77)
1367 (C 1 1)
1368 (T 6 136)
1369 (T 7 77)
1370 (C 22 1372)
1371 (U 5 18 22)
1372 (D 22 19)
1373 (C 21 1380)
1374 (J 1 137 21)
1375 (S 1)
1376 (S 136)
1377 (S 77)
1378 (C 21 1368)
1379 (J 0 0 21)
1380 (A 77 90)
1381 (C 1 1)
1382 (A 49 77)
1383 (S 77)
1384 (S 136)
1385 (T 6 136)
1386 (T 7 77)

1387 (C 22 1389)
1388 (U 5 18 22)
1389 (D 22 19)
1390 (S 137)
1391 (S 136)
1392 (S 77)
1393 (C 21 1398)
1394 (J 1 49 21)
1395 (S 1)
1396 (C 21 1385)
1397 (J 0 0 21)
1398 (T 6 100)
1399 (C 7 136)
1400 (C 22 1402)
1401 (U 5 18 22)
1402 (D 22 19)
1403 (T 136 100)
1404 (S 136)
1405 (C 5 2)
1406 (T 6 136)
1407 (C 7 137)
1408 (C 22 1410)
1409 (U 5 18 22)
1410 (D 22 19)
1411 (S 136)
1412 (T 6 136)
1413 (C 7 130)
1414 (C 22 1416)
1415 (U 5 18 22)
1416 (D 22 19)
1417 (S 136)
1418 (T 6 136)
1419 (C 7 140)
1420 (C 22 1422)
1421 (U 5 18 22)
1422 (D 22 19)
1423 (S 136)
1424 (T 6 136)
1425 (C 7 141)
1426 (C 22 1428)
1427 (U 5 18 22)
1428 (D 22 19)
1429 (S 136)
1430 (T 6 136)
1431 (C 7 132)
1432 (C 22 1434)
1433 (U 5 18 22)
1434 (D 22 19)
1435 (S 136)

1436 (T 6 136)
 1437 (C 7 133)
 1438 (C 22 1440)
 1439 (U 5 18 22)
 1440 (D 22 19)
 1441 (S 136)
 1442 (T 6 136)
 1443 (C 7 134)
 1444 (C 22 1446)
 1445 (U 5 18 22)
 1446 (D 22 19)
 1447 (T 93 94)
 1448 (S 93)
 1449 (S 93)
 1450 (S 93)
 1451 (S 93)
 1452 (S 93)
 1453 (S 93)
 1454 (S 93)
 1455 (C 3 0)
 1456 (S 136)
 1457 (S 93)
 1458 (T 6 136)
 1459 (T 7 93)
 1460 (C 22 1462)
 1461 (U 5 18 22)
 1462 (D 22 19)
 1463 (S 3)
 1464 (S 136)
 1465 (C 21 1469)
 1466 (J 3 97 21)
 1467 (C 21 1458)
 1468 (J 0 0 21)
 1469 (T 6 136)
 1470 (C 7 138)
 1471 (C 22 1473)
 1472 (U 5 18 22)
 1473 (D 22 19)
 1474 (S 101)
 1475 (C 151 1)
 1476 (C 0 0)
 1477 (C 21 1484)
 1478 (J 0 137 21)
 1479 (S 0)
 1480 (S 0)
 1481 (S 151)
 1482 (C 21 1477)
 1483 (J 0 0 21)
 1484 (C 150 0)

Instructions 1475 to 1537 look for a consecutive repeating state cycle
 in $(q_{0j}, a_{0j}), (q_{1j}, a_{1j}), \dots, (q_{NJ,j})$
 if $E([p_1, p_2, \dots, p_k, p_{k+1}], k+1)$ contains a consecutive repeating
 state cycle, as stated in 8.34, is executed in instructions 1475 to 1537

1485 (C 21 1538)
1486 (S 150)
1487 (J 150 151 21)
1488 (S 150)
1489 (J 150 151 21)
1490 (T 148 134)
1491 (S 148)
1492 (C 153 0)
1493 (T 149 148)
1494 (C 0 0)
1495 (C 21 1507)
1496 (J 0 150 21)
1497 (S 0)
1498 (S 0)
1499 (S 149)
1500 (S 149)
1501 (S 153)
1502 (S 153)
1503 (C 21 1485)
1504 (J 153 137 21)
1505 (C 21 1495)
1506 (J 0 0 21)
1507 (T 77 148)
1508 (T 78 149)
1509 (C 0 0)
1510 (T 154 153)
1511 (A 3 77)
1512 (A 4 78)
1513 (C 21 1517)
1514 (J 3 4 21)
1515 (C 21 1527)
1516 (J 0 0 21)
1517 (S 77)
1518 (S 78)
1519 (S 0)
1520 (S 154)
1521 (C 21 1557)
1522 (J 0 150 21)
1523 (C 21 1485)
1524 (J 154 137 21)
1525 (C 21 1511)
1526 (J 0 0 21)
1527 (S 148)
1528 (S 148)
1529 (S 149)
1530 (S 149)
1531 (C 21 1485)
1532 (S 153)
1533 (J 153 137 21)

1534	(S 153)	
1535	(J 153 137 21)	
1536	(C 21 1507)	
1537	(J 0 0 21)	
1538	(A 9 100)	
1539	(T 100 9)	
1540	(T 155 100)	
1541	(S 155)	
1542	(S 84)	
1543	(C 21 1031)	
1544	(J 0 0 21)	
1545	(S 96)	
1546	(A 95 94)	
1547	(T 94 95)	
1548	(C 21 1027)	
1549	(J 0 0 21)	
1550	(S 97)	
1551	(C 0 72)	Store "H" in register 0. ASCII code 72 = "H"
1552	(C 1 65)	Store "A" in register 1. ASCII code 65 = "A"
1553	(C 2 76)	Store "L" in register 2. ASCII code 76 = "L".
1554	(C 3 84)	Store "T" in register 3. ASCII code 84 = "T".
1555	(C 21 1590)	Put address 1590 in register 21 (JMP_ADDRESS)
1556	(J 0 0 21)	Jump to end of program. Halt.
1557	(C 0 73)	Store "I" in register 0. ASCII code 73 = "I"
1558	(C 1 77)	Store "M" in register 1. ASCII code 77 = "M"
1559	(C 2 77)	Store "M" in register 2. ASCII code 77 = "M"
1560	(C 3 79)	Store "O" in register 3. ASCII code 79 = "O"
1561	(C 4 82)	Store "R" in register 4. ASCII code 82 = "R"
1562	(C 5 84)	Store "T" in register 5. ASCII code 84 = "T".
1563	(C 6 65)	Store "A" in register 6. ASCII code 65 = "A"
1564	(C 7 76)	Store "L" in register 7. ASCII code 76 = "L".
1565	(T 8 150)	Store the length of one repeat of the consecutive repeating state cycle.
1566	(T 9 148)	Store the register where the first repeat of the state cycle begins.
1567	(T 10 149)	Store the register where the second repeat of the state cycle begins.
1568	(C 21 1590)	Put address 1590 in register 21 (JMP_ADDRESS)
1569	(J 0 0 21)	Jump to end of program. Halt.
1570	(C 0 69)	Store "E" in register 0. ASCII code 69 = "E"
1571	(C 1 82)	Store "R" in register 1. ASCII code 82 = "R"
1572	(C 2 82)	Store "R" in register 2. ASCII code 82 = "R"
1573	(C 3 79)	Store "O" in register 3. ASCII code 79 = "O"
1574	(C 4 82)	Store "R" in register 4. ASCII code 82 = "R"
1575	(C 5 32)	Store " " in register 5. ASCII code 32 = " "
1576	(C 6 84)	Store "T" in register 6. ASCII code 84 = "T".
1577	(C 7 85)	Store "U" in register 7. ASCII code 85 = "U".
1578	(C 8 82)	Store "R" in register 8. ASCII code 82 = "R"
1579	(C 9 73)	Store "I" in register 9. ASCII code 73 = "I"
1580	(C 10 78)	Store "N" in register 10. ASCII code 78 = "N"
1581	(C 11 71)	Store "G" in register 11. ASCII code 71 = "G"
1582	(C 12 32)	Store " " in register 12. ASCII code 32 = " "

1583	((C 13 80)	Store "P" in register 13. ASCII code 80 = "P"
1584	((C 14 82)	Store "R" in register 14. ASCII code 82 = "R"
1585	((C 15 79)	Store "O" in register 15. ASCII code 79 = "O"
1586	((C 16 71)	Store "G" in register 16. ASCII code 71 = "G"
1587	((C 17 82)	Store "R" in register 17. ASCII code 82 = "R"
1588	((C 18 65)	Store "A" in register 18. ASCII code 65 = "A"
1589	((C 19 77)	Store "M" in register 19. ASCII code 77 = "M"