CS & Problem Complexity

Tractable: - Algorithm Design & Analysis

$O(n^k)$, usually $k \leq 3$

Some seemingly hard problems are not so hard, but we approach in a way that repeats subproblems we already attacked -- memoization saves the day

Intractable: - Complexity Theory

$O(2^n)$ or worse, but still solvable given unbounded time and space. Also complexity of many problems solvable in $O(2^n)$ may not be inherently that hard.

Impossible: - Computability Theory

No algorithm can solve abitrary instances of problem

COMPUTABILITY

FOCUSES ON DECISION PROBLEMS (MOSTLY)

SIMILAR TO DFA OR NFA VERSUS TRANSDUCER (MEALY OR MOORE MACHINE)

HAS LONG HISTORY

RECENT (SORT OF) - 1900

HILBERT'S 23 PROBLEMS THAT WERE TO LEAD TO MECHANISM OF MATHEMATICAL REASONING & PROOFS

HILBERT'S 10TH (1900)

$P(\vec{x})=0$ $\vec{x}$ VECTOR OF INTEGRAL VARIABLES & P A POLYNOMIAL

FERMAT'S LAST THEOREM (1637)

$\exists n>2, a, b, c \in Z^+ \; [a^n = b^n + c^n]$

SHOWN TO HAVE NO SOLUTION IN [redacted] 1994/5

HILBERT'S 10TH SHOWN UNSOLVABLE IN (1971) BETTER PROOF IN 1973

ALGORITHMS AND PROCEDURES

PROCEDURES ARE JUST PROGRAMS OR PROCESSES THAT ARE CLEARLY COMPUTABLE.

PROCEDURES NEED NOT HALT FOR ALL INPUT

ALGORITHMS ARE PROCEDURES THAT HALT ON ALL INPUT

PREDICATES ARE PROCEDURES THAT PRODUCE ANSWERS (OUTPUT) TRUE/FALSE (YES/NO, 1/0)

DECISION PROBLEMS ARE PROBLEMS WHERE EACH INSTANCE HAS A TRUE/FALSE ANSWER

NOTATION:

$f(x)\downarrow$ MEANS PROCEDURE f HALTS (CONVERGES) GIVES AN OUTPUT) WHEN EVALUATED AT x

$f(x)\uparrow$ MEANS PROCEDURE f DIVERGES WHEN EVALUATED AT x

IF f IS AN ALGORITHM THEN $\forall x\, f(x)\downarrow$

COMPUTABILITY

SOLVED IS CONCRETE

SOLVABLE IS EXISTENTIAL

E.G., $P=NP?$ IS SOLVABLE

ANSWER IS "YES" OR "NO"

AND ALGORITHMS EXIST FOR EACH POSSIBLE ANSWER.

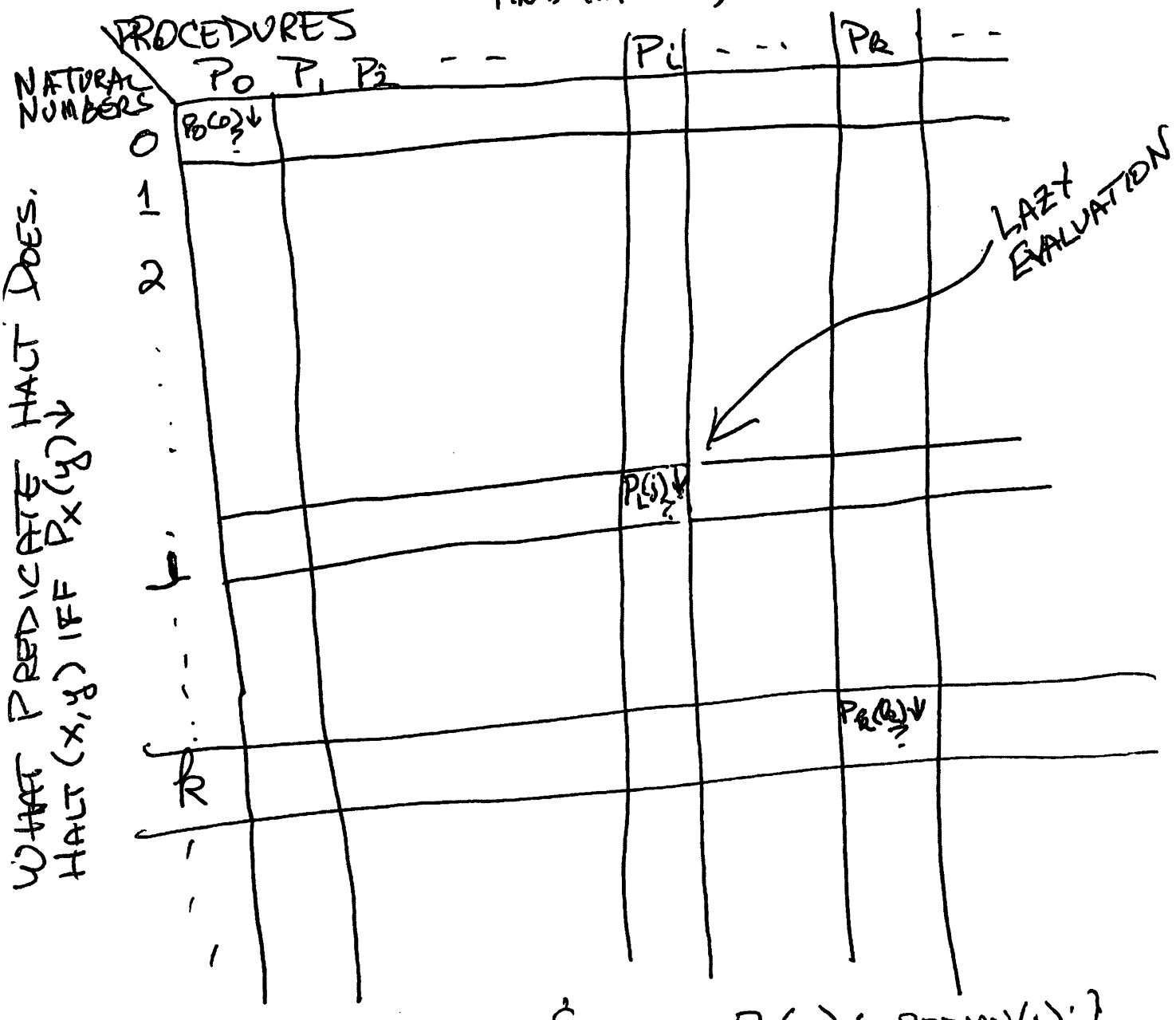
$P=NP?$ IS UNSOLVED AS NO ONE HAS PROVED THE PROPERTY TO BE SO, OR HAS REFUTED IT.

$\nexists\ a,b,c \left\{ a^n = b^n + c^n \right\}$

$a,b,c \in \mathbb{Z}^+,\ n>2$

HALTING PROBLEM

ASSUME WE CAN DECIDE FOR ARBITRARY PROCEDURE, P, AND INPUT X, WHETHER OR NOT P(X)↓



DEFINE $D(x) = \{$WHILE $P_x(x)$; RETURN(1);$\}$

$= \{$WHILE HALT(x,x); RETURN(1);$\}$

SINCE HALT IS AN ALGORITHM, IT IS A PROCEDURE AND HENCE SO IS D. ASSUME $D = P_d$ THEN

$D(d)$↓ IFF $P_d(d)$↓ IFF HALT(d,d) = TRUE IFF $P_d(d)$↑ IFF $D(d)$↑

$$U_{NIV}(p, x) = \varphi_p(x)$$

HALT PROBLEM

$$HALT(p,x) = \begin{cases} \text{TRUE} & \text{IF } \varphi_p(x)\downarrow \\ \text{FALSE} & \text{IF } \varphi_p(x)\uparrow \end{cases}$$

$$D(p) = \mu y\ HALT(p,p) = \text{FALSE}$$

$D(p)$ HALTS IFF $\varphi_p(p)\uparrow$

$\mu y\ U_{NIV}(p,p) == \text{FALSE}$

D IS A PROGRAM.

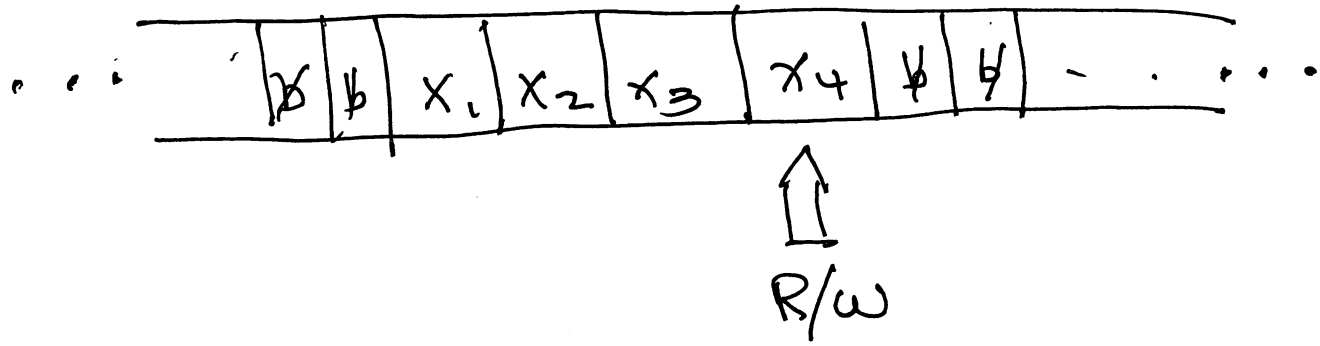
Assume $\varphi_d = D$

$D(d)\downarrow \iff HALT(d,d) == \text{FALSE}$

$\iff \varphi_d(d)\uparrow \iff D(d)\uparrow$

$\lightning$

TURING MACHINE



FINITE STATES $Q \times \Gamma$ DET.

POST MACHINE

$Q \times \Gamma \times \Gamma \times \{L, R\} \times Q$

WHAT I SEE AND MY STATE

WRITE

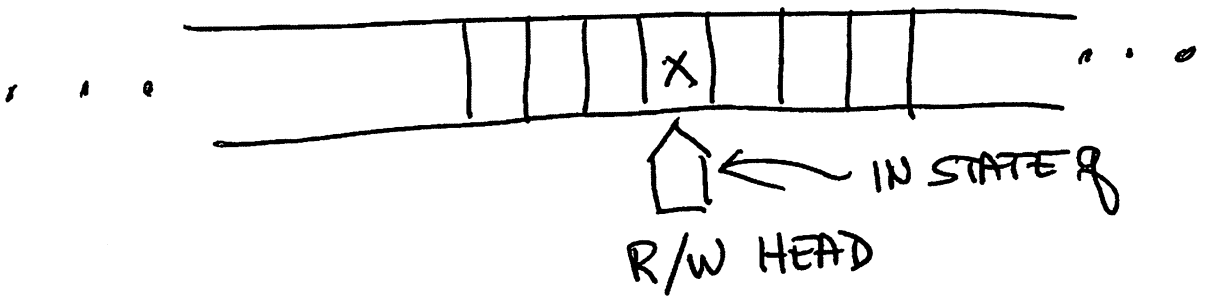
DIRECTION OF MOVEMENT

NEXT STATE

$Q \times \Gamma \times \{L, R\} \cup \Gamma \times Q$

MOVE OR WRITE

TURING MACHINE

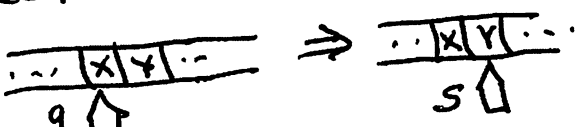


CAN READ CHARACTER IN SCANNED CELL (SAY X) AND BASED ON CURRENT STATE (SAY q) AND X (q, X IS CALLED THE DISCIMINANT)

CAN EITHER
(a) REWRITE X AS SOME OTHER SYMBOL;
(b) MOVE RIGHT; OR
(c) MOVE LEFT
AND THEN CHANGE STATE

NOTE: THIS IS REALLY POST'S NOTATION

q X R s



TM TAPE

UNMARKED PARTS OF TAPE ARE BLANK
TAPE STARTS WITH INPUT (FINITE)
AT EACH STAGE IT MIGHT MOVE TO
PARTS OF TAPE NEVER VISITED BEFORE
AND MAY WRITE NEW VALUES.

BASED ON THIS, TAPE IS ALWAYS
FINITELY MARKED AND ONLY A FINITE
NUMBER OF CELLS CAN BE VISITED
IN ANY FINITE PERIOD OF TIME

FOR OUR PURPOSES, WE WILL USE
TAPE ALPHABET OF $\{0,1\}$ AND
BLANK

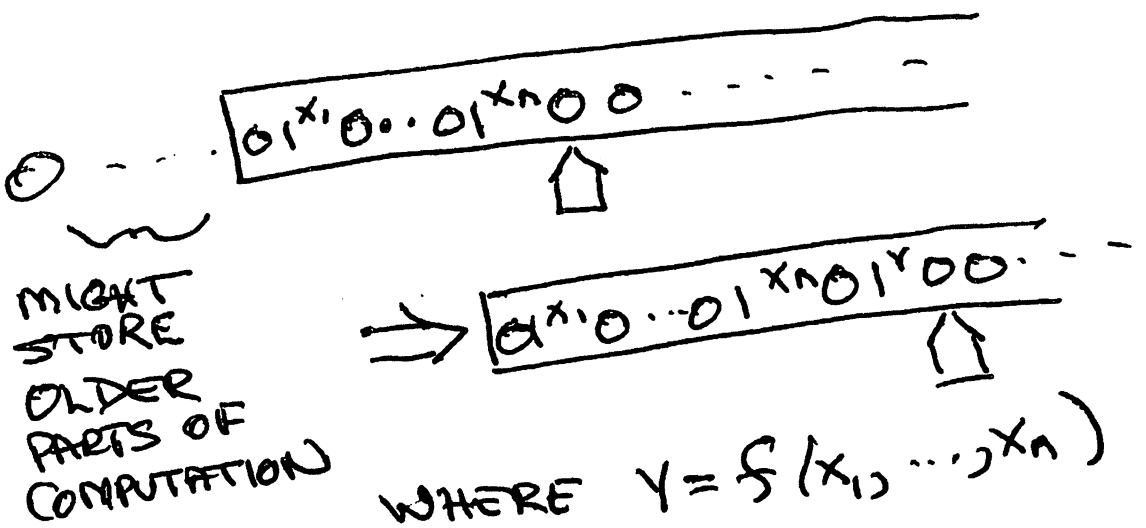
DENOTE NUMBERS IN UNARY.

$M = (Q, \{0,1\}, T)$ ← TABLE OF QUAD MAPPING

$Q \times \{0,1\} \to Q \times \{0,1,R,L\}$

HALTING IS JUST ENTERING A STATE
q WITH INPUT x, WHERE qx HAS NO
ENTRY IN T.

STANDARD TURING COMPUTING



WHERE $Y = f(x_1, \ldots, x_n)$

COMPOSITION IS EASY WITH THIS APPROACH TO COMPUTATION

SOMETIMES CALLED SEMI-UNBOUNDED TAPE

NOTE THAT TAPE IS ALWAYS FINITELY MARKED. CAN HAVE INSTANTANEOUS DESCR. (ID) AS TRIPLET (L, R, STATE) $\ldots0110\,10110\ldots$ $q_3 \rightarrow \langle 6,13,3 \rangle$

WHERE L IS # DENOTED BY BINARY VALUE OF LEFT OF SCANNED SQUARE;
R IS # DENOTED BY RIGHT AND SCANNED, READ RIGHT TO LEFT; STATE IS STATE #

INSANTANEOUS DESCRIPTION FOR TM

(i) AS CHARACTER STRING

$$\alpha q x \beta$$

α IS SHORTEST STRING COMPRISING ALL NON-BLANKS LEFT OF SCANNED SQUARE

q IS CURRENT STATE

x IS CHARACTER AT SCANNED SQUARE

β IS SHORTEST STRING COMPRISING ALL NON-BLANKS RIGHT OF SCANNED SQUARE

(ii) AS A TRIPLE OF INTEGERS (ASSUME 1 IS ONLY NON-BLANK; 0 IS BLANK)

$$[L, R, q]$$

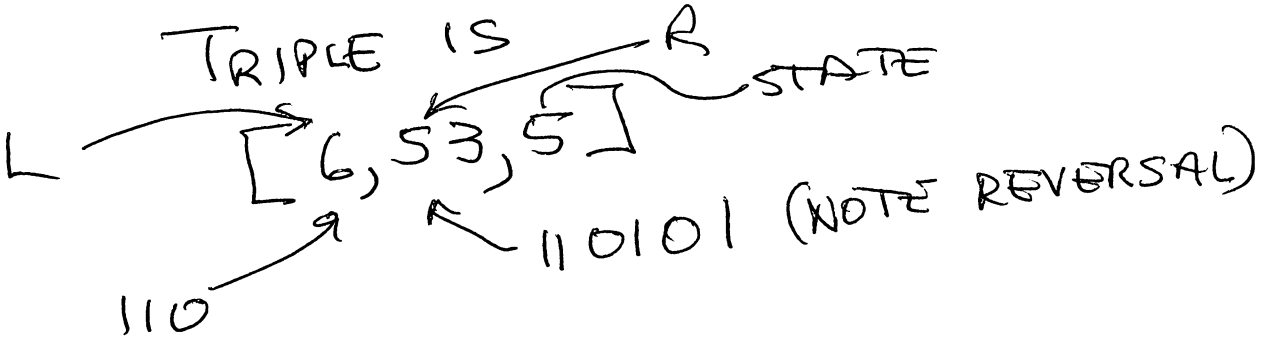
L IS NUMBER IN DECIMAL OF BINARY STRING TO LEFT OF SCANNED SQUARE

R IS NUMBER IN DECIMAL OF BINARY STRING THAT INCLUDES SCANNED SQUARE AND BIT TO RIGHT. IT IS READ IN REVERSE

EXAMPLE TM IDs

110q5101011

COULD BE A STRING REPRENTATION



AS TRIPLE, ODDNESS OF 53 MEANS WE ARE SCANNING A 1

IF TM HAS QUAD

$q_5\ 1\ R\ q_7$

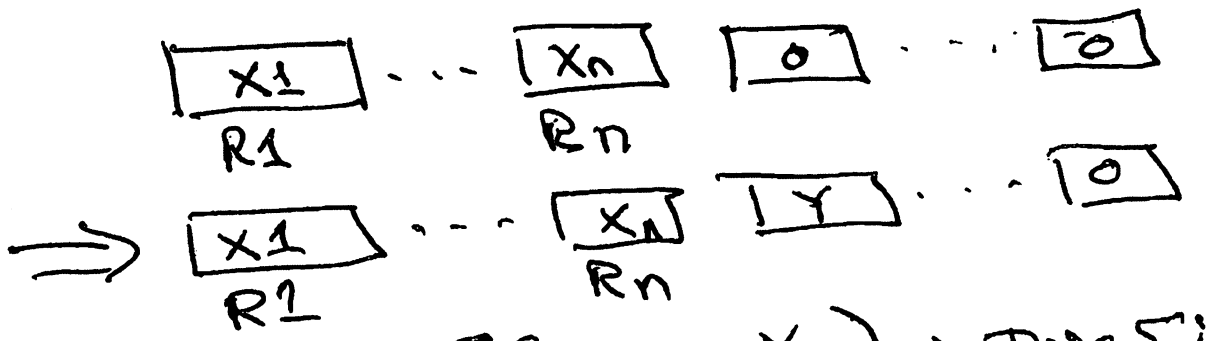
$[6,53,5] \vdash [13,26,7]$

IF $q_5\ 1\ L\ q_7$

$[6,53,5] \vdash [3,106,7]$

REGISTER MACHINES

MY STANDARD COMPUTATION



WHERE $Y = f(X_1, \ldots, X_n)$ $j. INC_R[i]$

$j. DEC_R[p,z]$ ↑ SUCCESS ↑ FAIL

ALTHOUGH WON'T SHOW, CAN MIMIC TM BY STORING ID IN 3 REGISTERS

CAN EVEN CREATE PROLOGUE THAT STARTS AS ABOVE AND CREATES THAT 3 REGISTER ENCODING

CAN HAVE EPILOGUE THAT EXTRACTS ANSWER FROM TRIPLE AT END OF SIMULATION

ID IS JUST

$\langle R1, R2, \ldots, R_k, STATE \rangle$ PAIRING

OR $3^{R1} 5^{R2} \ldots p_R^{R_k} p_{R+1}^{STATE}$

TM → RM

CAN STORE [L, R, q]
IN 3 REGISTERS, SAY, 1, 2, & 3

CAN DECREMENT R3 UNTIL ZERO
TO DETERMINE VALUE OF q.

CAN DECREMENT R2 IN PAIRED
DECREMENTS UNTIL ZERO TO DETERMINE
IF EVEN (SCANNED=0) OR ODD (SCANNED=1)
CAN SAVE & RESTORE THIS VALUE

IF q_0 Δ
SUBTRACT 1 FROM R2 IF ODD.

IF q_1 Δ
ADD 1 TO R2 IF EVEN.

IF q_R Δ
MULTIPLY R1 BY 2 AND ADD A 1 IF
R2 WAS ODD; DIVIDE R2 BY 2

IF q_L Δ
MULTIPLY R2 BY 2 AND ADD A 1 IF
R1 WAS ODD, DIVIDE R1 BY 2
ADD TO TO R3 TO GET IT TO VALUE

FACTOR REPLACEMENT SYSTEM

FINITE SEQUENCE (ORDERED) OF FRACTIONS

b_1/a_1
b_2/a_2
$\vdots$
b_k/a_k

$a_i, b_i \in Z^+$

$a_1 X \to b_1 X$
$a_2 X \to b_2 X$
$\vdots$
$a_k X \to b_k X$

} ALTERNATE FORM

STARTS WITH A NUMBER N

OPERATION
FIND LEAST i, a_i DIVIDES N
CHANGE N TO $N * b_i/a_i$
REPEAT UNTIL
(a) NO a_i DIVIDES N
OR (b) REACH A FIXED POINT

Factor Replacement System

$3^{x_1} \cdots P_k^{x_k} \Rightarrow 2^y$

$y = f(x_1, \ldots, x_k)$

$3^4 5^2$
$\Rightarrow 2 3^3 5^2$
$\Rightarrow 2^2 3^2 5^2$
⋮

$3x \to 2x$
$5x \to 2x$

$2^4 5^2$
$2^5 5$
2^6

2/3 OR
2/5 OR
$3^x 5^y \overset{*}{\Rightarrow} 2^{x+y}$

1/3.5 OR $15x \to x$
$3x \to 2x$
2/3
$5x \to x$
2/5 0/5
$3^x 5^y \overset{*}{\Rightarrow} 2^{MAX(x-y,0)}$

a_1/b_1 $\quad a_1 x \to a_1 x$
a_2/b_2
⋮
a_k/b_k $\quad b_k x \to a_k x$

IsPowerOf2

$3^{2.5}x \rightarrow 5.7x$ // ITERATIVELY DIV BY 2

$3.5.7x \rightarrow x$ // CAME UP ODD

$3.5x \rightarrow 2x$ // CASE OF $1=2^0$

$5.7x \rightarrow 7.11x$ // WAS DIV BY 2

$7.11x \rightarrow 3.11x$ // GET READY TO DIVIDE BY 2 AGAIN

$11x \rightarrow 5x$ // GO UP TO DIVIDE BY 2

$5x \rightarrow x$ // CASE OF 0

$7x \rightarrow x$ // CLEAN UP ODD CASE

$3^{2x}5 \rightarrow 7^x5 \rightarrow 7^x11 \rightarrow 3^x11 \rightarrow 3^x5$

$\ldots \rightarrow 3^1 5 \rightarrow 2=2^1$

$3^{2x+1}5 \rightarrow \ldots 37^x5 \rightarrow 7^x \rightarrow 1$

$35 \rightarrow 2=2^1 (3^1 5)$

$5 \rightarrow 1=2^0 (3^0 5)$

REGISTER TO FRS

REGISTERS
STORE n AS POWERS OF PRIMES
STORE STATE AS POWER OF NEXT PRIMES
PRESENCE IS EASY ($R_k > 0$) IFF

P_k IS A FACTOR

STATE CHECK IS EASY

P_{n+s} WHERE n REGISTERS

m. DEC k [i,j]

ORDER
$$\begin{cases} P_k P_{n+m} X \to P_{n+i} X \\ P_{n+m} X \to P_{n+j} X \end{cases}$$

m. INC k [i]

$P_{k+m} X \to P_k P_{n+i} X$