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:

CONVERGES

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

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

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

SOLVED

UNSOLVED

TEMPORAL

SOLVABLE
(RECURSIVE)
(DECIDABLE)

UNSOLVABLE
(NON-RECURSIVE)
(UNDECIDABLE)

INHERENT

SEMI-DECIDABLE
(ENUMERABLE)
(GENERABLE)
(RECURSIVELY ENUMERABLE)
(RE)

NOT SEMI-DECIDABLE
(NON-ENUMERABLE)
(NON-GENERABLE)
(NON-RE)

INHERENT

EXAMPLES

FERMAT'S LAST THEOREM IS SOLVED
AND ALWAYS WAS SOLVABLE

$$\nexists\, n > 2, a, b, c\ [a^n + b^n = c^n]$$

WHERE n, a, b, c ARE POSITIVE NATURAL NUMBERS

PROPERTY OF POLYNOMIAL TIME (DECISION) PROBLEMS
VERSUS NON-DET POLYNOMIAL TIME ONES

$$P \overset{?}{=} NP$$

IS UNSOLVED BUT SOLVABLE

PROPERTY TO DETERMINE, FOR ARBITRARY
PROGRAM, P, AND INPUT, X, WHETHER OR NOT
P(X) EVENTUALLY HALTS IS
UNSOLVABLE BUT SEMI-DECIDABLE

NOTE: $P(X)\downarrow$ MEANS P(X) CONVERGES

MORE EXAMPLES

PROPERTY TO DETERMINE, FOR ARBITRARY PROGRAM P, WHETHOR OR NOT

$\forall x\, P(x)\downarrow$ IS UNSOLVABLE AND NOT EVEN SEMI-DECIDABLE

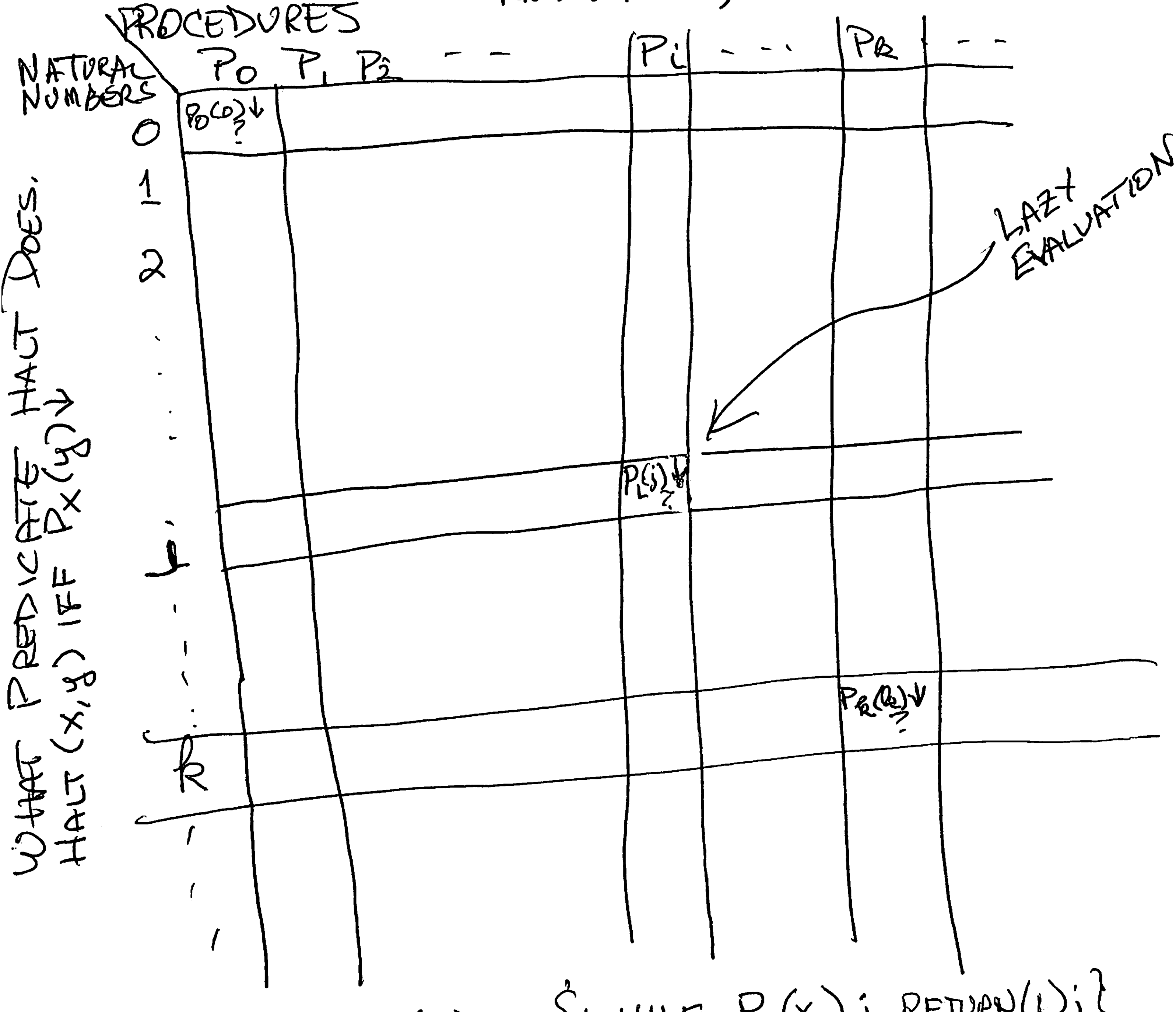
PROPERTY TO DETERMINE, FOR ARBITRARY POLYNOMIAL $P(x_1, \ldots, x_n)$ WHETHER OR NOT $\exists x_1, \ldots, x_n [P(x_1, \ldots, x_n) = 0]$ IS UNSOLVABLE BUT RE (SEMI-DECIDABLE)

CAN DO A HIGHLY STRUCTURED SEARCH (BREADTH FIRST) FOR A SOLUTION

LIMIT ON ALGORITHS TO A COUNTABLE SET MEANS THERE MUST BE SOME THAT CANNOT BE SOLVED OR EVEN ENUMERATED.

HALTING PROBLEM

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

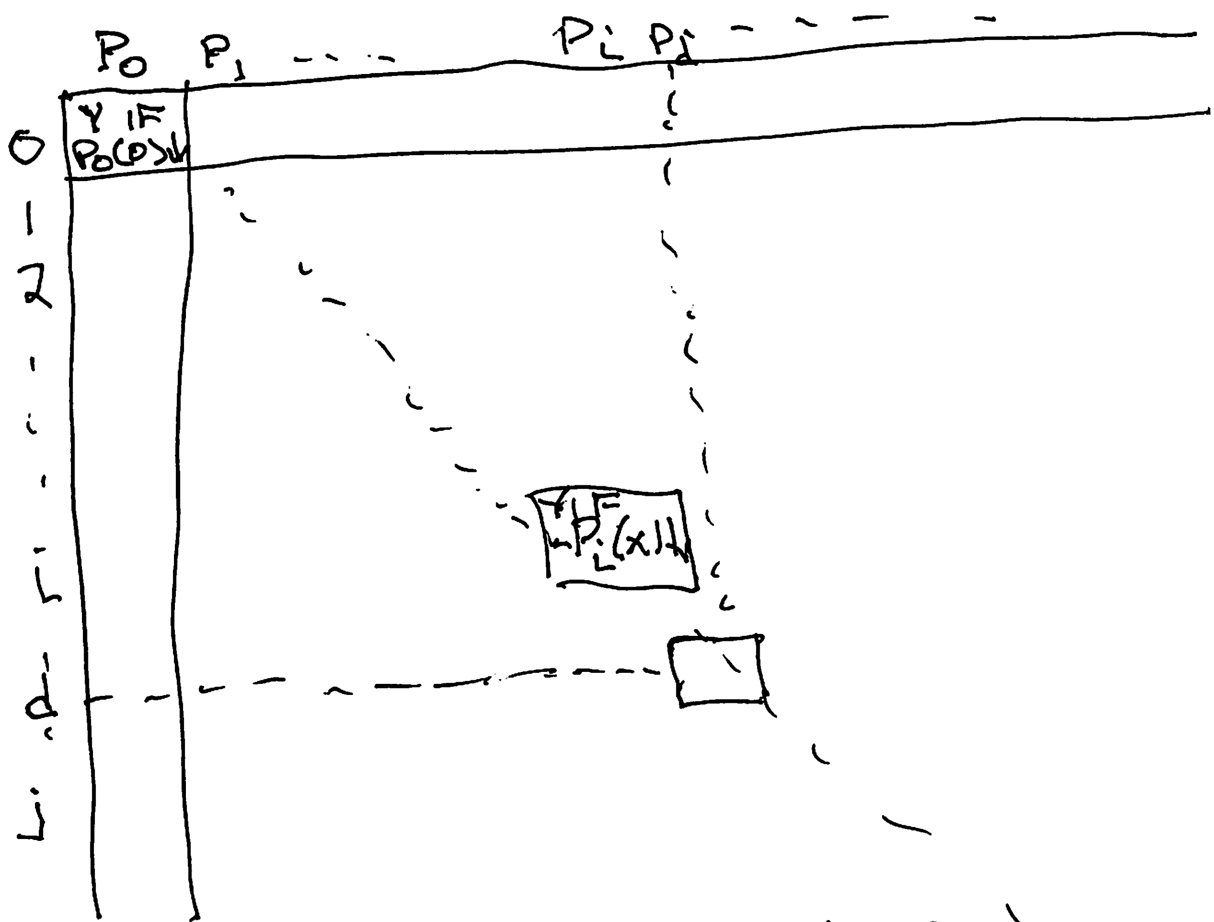


DEFINE $D(x) = \{\text{WHILE } P_x(x);\ \text{RETURN}(1);\}$
$= \{\text{WHILE HALT}(x,x);\ \text{RETURN}(1);\}$

SINCE HALT IS AN ALGORITHM, IT IS A PROCEDURE AND HENCE SO IS D. ASSUME $D = P_d$ THEN
$D(d)\downarrow$ IFF $P_d(d)\downarrow$ IFF HALT(d,d)=TRUE IFF
$P_d(d)\uparrow$ IFF $D(d)\uparrow$

ASSUME HALTING PROBLEM IS SOLVABLE

$$H(p,x) = \begin{cases} \text{YES} & \text{IF } p(x)\downarrow \\ \text{NO} & \text{IF } p(x)\uparrow \end{cases}$$



$$D(x) = \begin{cases} \downarrow & \text{IF } H(x,x) = \text{NO} \\ \uparrow & \text{IF } H(x,x) = \text{YES} \end{cases}$$

D(x):

```
    WHILE(H(x,x));
```

D IS A PROGRAM THEN $D = P_d$ FOR SOME d

$$D(d)\downarrow \Leftrightarrow H(d,d) = \text{NO} \Leftrightarrow P_d(d)\uparrow \Leftrightarrow D(d)\uparrow$$

WHAT CONTRADICTION REQUIRED

TO TALK ABOUT $P_0, P_1, \ldots$
WE NEED THAT WE CAN EFFECTIVELY MAP NATURAL NUMBERS TO PROGRAMS

IN CASE OF PROGRAMMING LANUAGES THAT IS JUST SORTING THEM, OR REALLY SORTING SYNTACTICALLY CORRECT PROGRAMS.

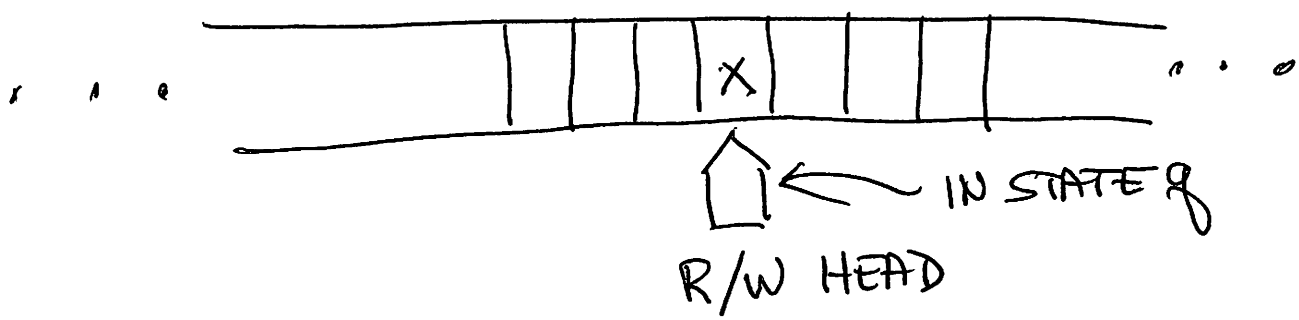
TO USE NOTATION $\mu y[y=y+1]$
WE JUST NEED A CONTROL CONSTRUCT THAT CAN RUN FOREVER.

COULD DEFINE D BY

```
D(i){
    WHILE (HALT(i,i));
    RETURN 0;
}
```

ASSUMING PREDICATE HALT EXISTS WHERE HALT(i,i) IFF $P_i(i)\downarrow$

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;

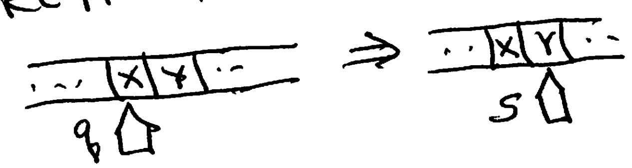
(b) MOVE RIGHT; OR

(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 $q\,x$ HAS NO ENTRY IN T. SO, REALLY $Q \times \{0,1\} \to Q \times \{0,1,R,L\} \cup \phi$

STANDARD TURING COMPUTING

$0 \cdots \boxed{0 1^{x_1} 0 \cdots 0 1^{x_n} 0\, 0 \cdots}$
$\Uparrow$

MIGHT STORE OLDER PARTS OF COMPUTATION

$\Rightarrow \boxed{0 1^{x_1} 0 \cdots 0 1^{x_n} 0 1^{y} 0\, 0 \cdots}$
$\Uparrow$

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)

$\ldots 0110\,10110 \ldots$
$q_3 \rightarrow \Uparrow \quad \langle 6, 13, 3 \rangle$

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 #

TM INST. DESCR. (ID)

TYPICALLY DONE WITH FINITE STRING

$\alpha q x \beta$

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

q IS CURRENT STATE

x IS SYMBOL BEING SCANNED

β IS SHORTEST STRING THAT ENCOMPASSES ALL NON-BLANKS RIGHT OF SCANNED

IF WE HAVE $\{0, 1\}$ TAPE ALPHABET
$\uparrow$ BLANK

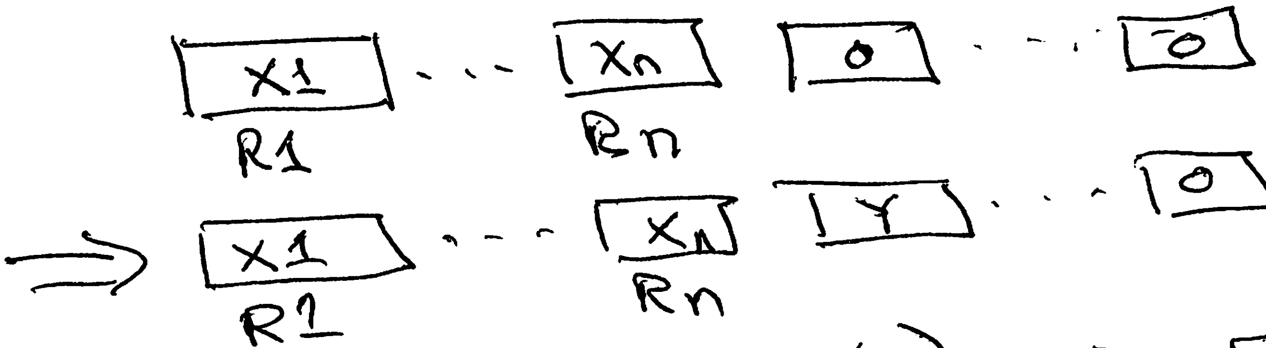
THEN ID CAN BE 3 NUMBERS

(L, R, STATE)

L IS α INTERPRETED AS BINARY NUMBER
R IS β^R " " " "
NOTE: IF $\alpha = \lambda$ OR $\beta = \lambda$, USE 0
STATE IS q

REGISTER MACHINES

MY STANDARD COMPUTATION



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

j. $INC_k[i]$

j. $DEC_k[p, z]$

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

OR $3^{R1} 5^{R2} \cdots P_k^{Rk} P_{k+1}^{STATE}$

FACTOR REPLACEMENT SYSTEM

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

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

$3^4 5^2$
$\Rightarrow 2 \cdot 3^3 5^2$
$\Rightarrow 2^2 3^2 5^2$
$\vdots$

$2/3$ OR $3x \to 2x$

$2/5$ OR $5x \to 2x$

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

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

$1/3 \cdot 5$ OR $15x \to x$

$2/3$ $3x \to 2x$

$2/5$ $1/5$ $5x \to x$

$3^x 5^y \overset{*}{\Rightarrow} 2^{\max(x-y,0)}$

a_1/b_1 $b_1 x \to a_1 x$

a_2/b_2

$\vdots$

a_k/b_k $b_k x \to a_k x$

REGISTER TO FRS

STORE REGISTERS AS POWERS OF PRIMES

STORE STATE AS POWER OF NEXT PRIMES

PRESENCE IS EASY $(R_k > 0)$

P_k

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]

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

IsPowerOf 2

$3^2 \cdot 5x \to 5 \cdot 7x$ // ITERATIVELY DIV BY 2

$3 \cdot 5 \cdot 7x \to x$ // CAME UP ODD

$3 \cdot 5x \to 2x$ // CASE OF $1 = 2^0$

$5 \cdot 7x \to 7 \cdot 11x$ // WAS DIV BY 2

$7 \cdot 11x \to 3 \cdot 11x$ // GET READY TO DIVIDE BY 2 AGAIN

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

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

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

$$3^{2x} 5 \to 7^x 5 \to 7^x 11 \to 3^x 11 \to 3^x 5$$

$$\ldots \; 3^1 5 \to 2 = 2^1$$

$$3^{2x+1} 5 \to \ldots \; 3 7^x 5 \to 7^x \to 1$$

$$3\,5 \to 2 = 2^1 \; (3^1 5)$$

$$5 \to 1 = 2^0 \; (3^0 5)$$