EQUIVALENCE

$$TM \leq RM \leq FRS \leq REC \leq TM$$

UNARY ALPHABET WITH 0 AS BLANK

REPRESENTING WORDS OVER LARGER ALPHABETS

$\Sigma = \{a, b, c\}$

WORD = acab

$0\underbrace{0101110101100}\,0$

00 SEPARATES WORDS

THUS, WE CAN FOCUS ON TAPE ALPHABET OR $\{1\}$ WITH BLANK AS 0.

ENCODING TM INSTANTANEOUS DESCRIPTION

STRING APPROACH

$$\ldots 00\underbrace{1010011}\,q_7\,\underbrace{\underline{0}1}\,00\ldots$$

$1010011\,q_7\,01$

RECORD SHORTEST STRING ON RIGHT THAT INCLUDES SCANNED SQUARE AS RIGHTMOST NON-BLANK

RECORD SHORTEST STRING ON LEFT THAT INCLUDES LEFTMOST NON-BLANK

PLACE STATE # TO LEFT OF SCANNED SQUARE

INTEGER APPROACH

$(2, 83, 7)$ FOR $1010011\,q_7\,01$

RIGHT READ R TO L

LEFT READ L TO R

STATE INDE

NOTE:

IF FIRST NUMBER IS EVEN, SCANNED SQUARE IS 0; IF ODD, THEN 1.
SAME FOR RIGHTMOST SYMBOL ON LEFT

TM ≤ REGISTER MACHINE

CAN STORE TM ID IN JUST THREE REGISTERS

CAN SHIFT LEFT VIA MULTIPLY BY 2

ASSUME $r_2 = 0$; $r_3 = 0$

$$\left.\begin{array}{ll} x. & \mathrm{DEC}_{r_1}(x+1, x+4) \\ x+1. & \mathrm{INC}\ r_2\ (x+2) \\ x+2. & \mathrm{INC}\ r_2\ (x+3) \\ x+3. & \mathrm{INC}\ r_3\ (x) \end{array}\right\} \begin{array}{l} r_2 = r_1 * 2 \\ r_3 = r_1 \\ r_1 = 0 \end{array}$$

$$\left.\begin{array}{ll} x+4. & \mathrm{DEC}_{r_3}(x+5, x+6) \\ x+5. & \mathrm{INC}_{r_1}(x+4) \end{array}\right\} \begin{array}{l} r_1 = r_3 \\ r_3 = 0 \end{array}$$

$x+6.$

CAN SHIFT RIGHT VIA DIVIDE BY 2

DETAILS OF TM ≤ RM

IN SUPPLEMENTAL NOTES

RM ≤ FRS

ID FOR RM IS

$$p_1^{r_1}\, p_2^{r_2} \cdots p_n^{r_n}\, p_{n+j}$$

WHERE r_k IS CONTENTS OF REGISTER k
AND WE ARE ABOUT TO EXECUTE INSTR. j.

CAN SIMULATE BY

j. $\mathrm{INCR}_r[i]$

$$p_{n+j}\, x \rightarrow p_{n+i}\, p_r\, x$$

j. $\mathrm{DEC}_r[s,f]$

$$p_{n+j}\, p_r\, x \rightarrow p_{n+s}\, x$$
$$p_{n+j}\, x \rightarrow p_{n+f}\, x$$

ALSO

$$p_{n+m+1}\, x \rightarrow x$$

FOR HALTING CONDITION

DETAILS IN SUPPLEMENTAL NOTES

Universal Machine

- In the process of doing this reduction, we will build a Universal Machine.
- This is a single recursive function with two arguments. The first specifies the factor system (encoded) and the second the argument to this factor system.
- The Universal Machine will then simulate the given machine on the selected input.

Encoding FRS

- Let $(\mathbf{n}, ((\mathbf{a}_1,\mathbf{b}_1), (\mathbf{a}_2,\mathbf{b}_2), \ldots ,(\mathbf{a}_n,\mathbf{b}_n))$ be some factor replacement system, where $(\mathbf{a}_i,\mathbf{b}_i)$ means that the **i**-th rule is

  $$\mathbf{a}_i\mathbf{x} \quad \rightarrow \quad \mathbf{b}_i\mathbf{x}$$

- Encode this machine by the number **F**,

  $$2^n 3^{a_1} 5^{b_1} 7^{a_2} 11^{b_2} \cdots p_{2n-1}^{a_n} p_{2n}^{b_n} p_{2n+1} p_{2n+2}$$

Simulation by Recursive # 1

- We can determine the rule of **F** that applies to **x** by

  RULE(F, x) = μ z (1 ≤ z ≤ exp(F, 0)+1) [exp(F, 2*z-1) | x]

- Note: if **x** is divisible by a_i, and **i** is the least integer for which this is true, then **exp(F,2*i-1) =** a_i where a_i is the number of prime factors of **F** involving p_{2i-1}. Thus, **RULE(F,x) = i**.

  If x is not divisible by any a_i, $1 \le i \le n$, then **x** is divisible by **1**, and **RULE(F,x)** returns **n+1**. That's why we added $p_{2n+1}\, p_{2n+2}$.

- Given the function **RULE(F,x)**, we can determine **NEXT(F,x)**, the number that follows **x**, when using **F**, by

  NEXT(F, x) = (x // exp(F, 2*RULE(F, x)-1)) * exp(F, 2*RULE(F, x))

Simulation by Recursive # 2

- The configurations listed by **F**, when started on **x**, are

CONFIG(F, x, 0) = x

CONFIG(F, x, y+1) = NEXT(F, CONFIG(F, x, y))

- The number of the configuration on which **F** halts is

HALT(F, x) = μ y [CONFIG(F, x, y) == CONFIG(F, x, y+1)]

This assumes we converge to a fixed point only if we stop

Simulation by Recursive # 3

- A Universal Machine that simulates an arbitrary Factor System, Turing Machine, Register Machine, Recursive Function can then be defined by
  Univ(F, x) = exp (CONFIG (F, x, HALT (F, x)), 0)
- This assumes that the answer will be returned as the exponent of the only even prime, **2**. We can fix **F** for any given Factor System that we wish to simulate.

Example of Universal Machine in Action

$$3^a 5^b \overset{*}{\Rightarrow} 2^{|a-b|} \qquad \begin{array}{l} 3 \cdot 5x \to x \\ 3x \to 2x \\ 5x \to 2x \end{array}$$

$$F = 2^3 \overbrace{3^{15} 5^1}^{3.5x \to x} \overbrace{7^3 11^2}^{3x \to 2x} \overbrace{13^5 19^2}^{5x \to 2x} \underbrace{19^1 23^1}_{}$$

IMPLICIT $x \to x$

Do for $x = 3^2 5^4$

$\text{Rule}(F, x) = \mu z (1 \le z \le 4) [p_{2z-1} \mid x]$

$\text{Rule}(F, 3^2 5^4) = 1$

$\text{Rule}(F, 3^2 5^4) = (3^2 5^4 // 15) * 1 = 3^1 5^3$

$\text{Next}(F, 3^2 5^4) =$

$\text{Rule}(F, 3^1 5^3) = 1$

$\text{Rule}(F, 3^1 5^3) = (3 5^3 // 15) * 1 = 5^2$

$\text{Next}(F, 3^1 5^3) =$

$\text{Rule}(F, 5^2) = 3$

$\text{Rule}(F, 5^2) = (5^2 // 5) * 2 = 2^1 5^1$

$\text{Next}(F, 5^2) = (2^1 5^1 // 5) * 2 = 2^2$

$\text{Next}(F, 2^1 5^1) =$

$\text{Next}(F,$

$\text{Rule}(F, 2^1 5^1) = 3$

$\text{Rule}(F, 2^2) = 4$

$\text{Rule}(F, 2^2) = (2^2 // 1) * 1 = 2^2$

$\text{Next}(F, 2^2) =$

$\text{Next}(F, 3^2 5^4, 0) = 3^2 5^4$

$\text{Config}(F, 3^2 5^4, 1) = 3^1 5^3$

$\text{Config}(F, 3^2 5^4, 2) = 5^2$

$\text{Config}(F, 3^2 5^4, 3) = 2^1 5^1$

$\text{Config}(F, 3^2 5^4, 4) = 2^2$

$\text{Config}(F, 3^2 5^4, 5) = 2^2$ $\Big\}$ $\text{Halt}(F, 3^2 5^4) = 4$

$\text{Config}(F, 3^2 5^4, 5)$

RESULT FOR EXAMPLE

AGAIN,

$$\text{HALT}(F, 3^2 5^4) = 4$$

SO,

$$\text{UNIV}(F, 3^2 5^4) = \text{EXP}(\text{CONFIG}(F, 3^2 5^4, 4), 0)$$
$$= \text{EXP}(2^2, 0)$$
$$= 2$$

NOTE: F AND X WERE ARBITRARY EXCEPT THAT F WAS A FRS AND X WAS LEGIT INPUT ENCODING. WE COULD WRITE RECURSIVE FUNCTIONS THAT SYNTACTICALLY CHECK F AND X, OR EVEN JUST CHECKING F WORKS

RECURSIVE ≤ TURING

SHOW BASE FUNCTIONS ARE TURING COMPUTABLE

$C_a^n(x_1, \dots, x_n) = a$

$(R\,1)^a R$

$I_i^n(x_1, \dots, x_n) = x_i$

C_{n-i+1}

$S(x) = x+1$

$C_1\, 1\, R$

NOW SHOW TURING COMPUTABLE CLOSED UNDER COMPOSITION, INDUCTION AND MINIMIZATION

DETAILS IN SUPPLEMENTAL NOTES

UNIVERSAL MACHINE

REALLY AN INTERPRETER FOR PROGRAMS IN SOME MODEL OF COMPUTATION, WRITTEN IN THAT MODEL

$$\text{UNIV}(x,y) = \varphi_x(y)$$

WHERE φ_x IS X-TH PROGRAM IN SOME WAY OF ORDERING PROGRAMS, E.G., LEXICALLY.

$$\varphi(x,y) = \text{UNIV}(x,y) = \varphi_x(y)$$

HALTING PROBLEM

ASSUME ALGORITHMIC PREDICATE HALT

$$\text{HALT}(f, x) \Leftrightarrow \varphi_f(x)\downarrow$$

DEFINE

$$\text{DISAGREE}(x, \cancel{x}) = \mu y\, [\sim \text{HALT}(x,x)]$$

NOT

CLEARLY

IF $\sim \text{HALT}(x,x)$ THEN $\text{DISAGREE}(x) = 0$

IF $\cancel{\sim}\, \text{HALT}(x,x)$ THEN $\text{DISAGREE}(x)\uparrow$

OR $\quad \text{HALT}(x,x) \Leftrightarrow \text{DISAGREE}(x)\uparrow$

OR $\quad \varphi_x(x)\downarrow \Leftrightarrow \text{DISAGREE}(x)\uparrow$

SINCE HALT IS AN ALGORITHM, DISAGREE IS AN EFFECTIVE PROCEDURE AND SO, FOR SOME d,

$$\varphi_d = \text{DISAGREE}$$

BUT THEN

$$\varphi_d(d)\downarrow \Leftrightarrow \text{DISAGREE}(d)\uparrow \Leftrightarrow \varphi_d(d)\uparrow$$

SO HALT CANNOT EXIST