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 = $a\,c\,a\,b$

$0\underbrace{0101110101100}\,0$

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
INDEX

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 $r_2 = 0$; $r_3 = 0$

X. $DEC_{r_1}(X+1, X+4)$
X+1. $INC\ r_2\ (X+2)$
X+2. $INC\ r_2\ (X+3)$
X+3. $INC\ r_3\ (X)$

$r_2 = r_1 * 2$
$r_3 = r_1$
$r_1 = 0$

X+4. $DEC_{r_3}(X+5, X+6)$
X+5. $INC_{r_1}(X+4)$

$r_1 = r_3$
$r_3 = 0$

X+6.

CAN SHIFT RIGHT VIA DIVIDE BY 2

DETAILS OF TM ≤ RM ON NOTES 488-494

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 ON PAGES 495–501

$R_0^0 R_1^X R_2^Y R_3^0 R_4^0 \vdash^* R_0^{X*Y} R_1^X R_2^Y R_3^0 R_4^0$

1. DEC_2 [2,8]
2. INC_4 [3]
3. DEC_1 [4,6]
4. INC_0 [5]
5. INC_3 [3]
6. DEC_3 [7,1]
7. INC_1 [6]
8. DEC_4 [9,10]
9. INC_2 [8]
10.

$13 \cdot 5x \to 17x ;\ 13x \to 41x$

$17x \to 19 \cdot 11x$

$19 \cdot 3x \to 23x ;\ 19x \to 31x$

$23x \to 29 \cdot 2x$

$29x \to 19 \cdot 7x$

$31 \cdot 7x \to 37x ;\ 31x \to 13x$

$37x \to 31 \cdot 3x$

$41 \cdot 11x \to 43x ;\ 41x \to x$

$43x \to 41 \cdot 5x$

$13 \cdot 3^X \cdot 5^Y \vdash^* 2^{X*Y} 3^X 5^Y$

OR COULD END UP AT $2^{X*Y} 3^X 5^Y \cdot 47$

OR EVEN 2^{X*Y}

OR

STATE	PRIME
1	13
2	17
3	19
4	23
5	27
6	31
7	37
8	41
9	43

REGISTER	PRIME
0	2
1	3
2	5
3	7
4	11

$$13 \cdot 3^X 5^Y \vdash^* 2^{X*Y} 3^X 5^Y$$

$13 \cdot 5x \to 17x$

$13x \to 41x$

$17x \to 19 \cdot 11x$

$19 \cdot 3x \to 23x$

$19x \to 31x$

$23x \to 29 \cdot 2x$

$29x \to 19 \cdot 7x$

$31 \cdot 7x \to 37x$

$31x \to 13x$

$37x \to 31 \cdot 3x$

$41 \cdot 11x \to 43x$

$41x \to x$

$43x \to 41 \cdot 5x$

STATE	PRIME
1	13
2	17
3	19
4	23
5	29
6	31
7	37
8	41
9	43

REGISTER	PRIME
0	2
1	3
2	5
3	7
4	11

Importance of Order

- The relative order of the two rules to simulate a **DEC** are critical.
- To test if register **r** has a zero in it, we, in effect, make sure that we cannot execute the rule that is enabled when the **r**-th prime is a factor.
- If the rules were placed in the wrong order, or if they weren't prioritized, we would be non-deterministic.

Example of Order

Consider the simple machine to compute **r1:=r2 – r3** (limited)

1. **DEC3[2,3]**
2. **DEC2[1,1]**
3. **DEC2[4,5]**
4. **INC1[3]**
5.

Subtraction Encoding

Start with $\mathbf{3^x5^y7}$

7 • 5 x	$\rightarrow$	**11 x**
7 x	$\rightarrow$	**13 x**
11 • 3 x	$\rightarrow$	**7 x**
11 x	$\rightarrow$	**7 x**
13 • 3 x	$\rightarrow$	**17 x**
13 x	$\rightarrow$	**19 x**
17 x	$\rightarrow$	**13 • 2 x**
19 x	$\rightarrow$	**x**

Analysis of Problem

- If we don't obey the ordering here, we could take an input like $\mathbf{3^5 5^2 7}$ and immediately apply the second rule (the one that mimics a failed decrement).
- We then have $\mathbf{3^5 5^2 13}$, signifying that we will mimic instruction number **3**, never having subtracted the **2** from **5**.
- Now, we mimic copying **r2** to **r1** and get $\mathbf{2^5 5^2 19}$.
- We then remove the **19** and have the wrong answer.

FACTOR $\leq$ RECURSIVE

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 **(n, ((a_1,b_1), (a_2,b_2), … ,(a_n,b_n))** be some factor replacement system, where **(a_i,b_i)** means that the **i**-th rule is

  $$\mathbf{a_i x} \quad \rightarrow \quad \mathbf{b_i x}$$

- 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

  $$\mathbf{RULE(F, x) = \mu\, z\, (1 \le z \le exp(F, 0)+1)\, [\, exp(F, 2*z-1) \mid x\,]}$$

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

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

FRS Subtraction

- $2^0 3^a 5^b \Rightarrow 2^{a-b}$
  $3*5x \rightarrow x$ or 1/15
  $5x \rightarrow x$ or 1/5
  $3x \rightarrow 2x$ or 2/3
- Encode $F = 2^3\ 3^{15}\ 5^1\ 7^5\ 11^1\ 13^3\ 17^2\ 19^1\ 23^1$
- Consider a=4, b=2
- RULE(F, x) = μ z ($1 \leq z \leq 4$) [exp(F, 2*z-1) | x]
  RULE (F,$3^4\ 5^2$) = 1, as 15 divides $3^4\ 5^2$
- NEXT(F, x) = (x // exp(F, 2*RULE(F, x)-1)) * exp(F, 2*RULE(F, x))
  NEXT(F,$3^4\ 5^2$) = ($3^4\ 5^2$ // 15 * 1) = $3^3 5^1$
  NEXT(F,$3^3\ 5^1$) = ($3^3\ 5^1$ // 15 * 1) = 3^2
  NEXT(F,3^2) = (3^2 // 3 * 2) = $2^1 3^1$
  NEXT(F, $2^1 3^1$) = ($2^1 3^1$ // 3 * 2) = 2^2
  NEXT(F, 2^2) = (2^2 // 1 * 1) = 2^2

Rest of simulation

- CONFIG(F, x, 0) = x
  CONFIG(F, x, y+1) = NEXT(F, CONFIG(F, x, y))
- CONFIG(F,3^4 5^2,0) = 3^4 5^2
  CONFIG(F,3^4 5^2,1) = $3^3 5^1$
  CONFIG(F,3^4 5^2,2) = 3^2
  CONFIG(F,3^4 5^2,3) = $2^1 3^1$
  CONFIG(F,3^4 5^2,4) = 2^2
  CONFIG(F,3^4 5^2,5) = 2^2
- HALT(F, x)=μy[CONFIG(F,x,y)==CONFIG(F,x,y+1)] = 4
- Univ(F, x) = exp (CONFIG (F, x, HALT (F, x)), 0)
  = exp(2^2,0) = 2

Simplicity of Universal

- A side result is that every computable (recursive) function can be expressed in the form

$$\mathbf{F(x) = G(\mu\ y\ H(x, y))}$$

where **G** and **H** are primitive recursive.

RECURSIVE $\leq$ 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 ON NOTES PAGES 511–518

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.

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

HALTING PROBLEM

<u>RE</u> BUT NOT SOLVABLE

LET f BE INDEX OF SOME ARBITRARY PROCEDURE (PROGRAMS CAN BE ORDERED)

LET x BE AN ARBITRARY MEMBER OF $\mathbb{N}$

REALLY BOTH f AND x ARE IN $\mathbb{N}$

WANT TO DECIDE IF $\varphi_f(x)\downarrow$

CAN EASILY SEMI-DECIDE BY RUNNING

$$\varphi(f,x)$$

WHEN HALTS, RETURN TRUE, ELSE RUNS FOREVER. HOWEVER, THIS IS NOT DECIDABLE AS WE SEE ON NEXT PAGE.

NOTE:

$$SDHALT(f,x) = (\varphi(f,x) == \varphi(f,x))$$

IF FAILS TO CONVERGE THEN $SDHALT(f,x)\uparrow$

$$Dom(SDHALT) = \{\langle f,x\rangle \mid \varphi_f(x)\downarrow\}$$

HALTING PROBLEM

ASSUME THERE EXISTS AN ALGORITHM HALT

SUCH THAT $HALT(x, y) = 1$ IF $\varphi_x(y)\downarrow$

$= 0$ OTHERWISE

DEFINE DISAGREE BY

$$DISAGREE(x) = \mu y [HALT(x, x) == 0]$$

NOTE: IF $HALT(x, x) = 0$ THEN

$DISAGREE(x) = 0$

IF $HALT(x, x) = 1$ THEN

$DISAGREE(x)\uparrow$ (DIVERGES)

AS DISAGREE IS CLEARLY A μ-RECURSIVE FUNCTION, IT IS A COMPUTABLE FUNCTION AND IT HAS AN INDEX, CALL IT d. $\varphi_d = D$

BUT $D(d)\downarrow$ IFF $HALT(d, d) = 0$

IFF $\varphi_d(d)\uparrow$

IFF $D(d)\uparrow$

CONTRADICTION. THUS, HALT CANNOT EXIST

ENUMERATION THEOREM

DEFINE

$$W_n = \{x \in \mathbb{N} \mid \varphi(n,x)\downarrow\}$$

φ_n SEMI-DECIDES W_n SO

THEOREM: A SET B IS RE IFF $\exists n$ SUCH THAT $B = W_n$

THIS ALLOWS US TO ENUMERATE THE RE (SEMI-DECIDABLE) SETS

UNIFORM HALTING PROBLEM

AKA TOTAL φ_f f

$\text{TOTAL} = \{ f \mid \varphi_f \text{ IS AN ALGORITHM} \}$

$= \{ f \mid \forall x\, \varphi_f(x)\downarrow \}$ $\forall x\, \varphi_f(x)\downarrow$

$\forall x\ f(x)\downarrow$

ASSUME TOTAL IS RE AND A IS AN ENUMATING ALGORITHM

$A(x) = A_x$

WHERE $A_0, A_1, A_2, \ldots$ IS LIST OF INDICES OF ALL AND ONLY ~~THE~~ ALGORITHMS

DEFINE $G(x) = \text{Univ}(A(x), x) + 1$

$= \varphi_{A(x)}(x) + 1$

BUT THEN G IS AN ALGORITHM, SAY THE g-TH ONE

THAT IS, $A(g) = A_g = G$

THUS, $G(g) = \varphi_{A(g)}(g) + 1 = G(g) + 1$

BUT THAT IS A CONTRADICTION SINCE G IS AN ALGORITHM

NOTE: IF G WERE A PROCEDURE, THIS IS NOT NEC. A CONTRADICTION — WHY?

Input \ Algorithms	A_0	A_1	...	A_k	...
0	$A_0(0)+1$	$A_1(0)$		$A_k(0)$	
1	$A_0(1)$	$A_1(1)+1$		$A_k(1)$	
$\vdots$					
k	$A_0(k)$	$A_1(k)$		$A_k(k)+1$	
$\vdots$					

THIS IS POSSIBLE IF CAN LIST (REC. ENUM.) THE ALGORITHMS (SUBSET OF PROCEDURES, WHICH ARE ENUMERABLE).

MORE ON TOTAL

$$\text{TOTAL} = \{f \in N \mid \forall x\, \varphi_f(x)\downarrow\}$$
$$= \{f \in N \mid \omega_f = N\}$$

OUR RESULT THAT TOTAL IS NOT RE MEANS THERE IS NO COMPLETE MODEL OF COMPUTATION THAT DOES NOT INCLUDE PROCEDURES THAT ARE NOT ALGORITMS.

THAT IS, NO GENERATIVE SYSTEM (E.G., GRAMMAR) CAN PRODUCE DESCRIPTIONS OF ALL AND ONLY ALGORITHMS

<u>AND</u>

NO PARSING SYSTEM CAN ACCEPT ALL AND ONLY ALGORITHMS

THAT IS, REAL COMPUTER LANGUAGES MUST SUPPORT INFINITE COMPUTATION (LOOPS)

WAYS TO DIVERGE

WHILE LOOPS

GOTO'S AS IN TMs AND RMs

MINIMIZATION AS IN REC FUNCTIONS

FIXED POINT AS IN FRS

NON-DETERMINISM IN MODELS

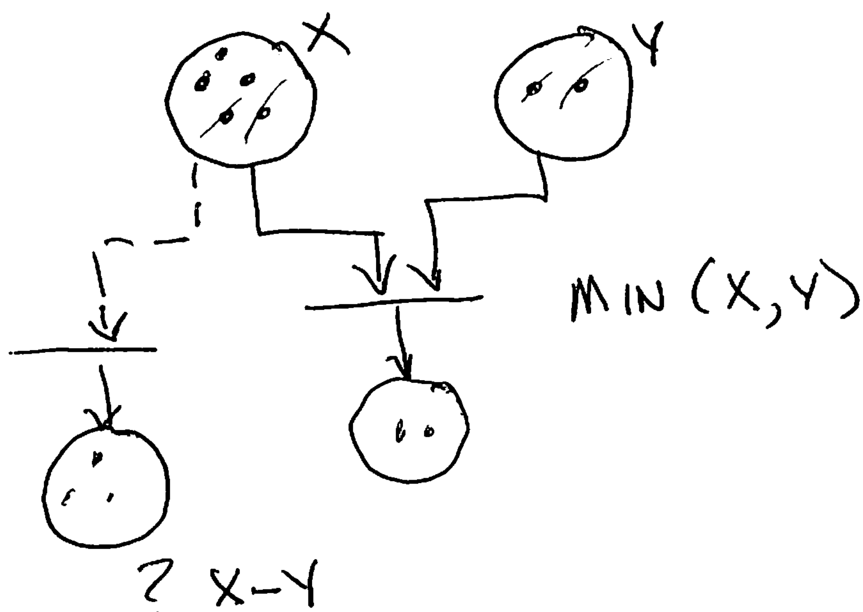
HELPS FOR PDAs

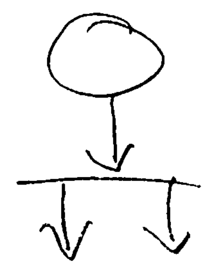
DOESN'T HELP OR HURT FOR

- FINTE STATE AUTOMATA
- LBAs
- TURING MACHINES

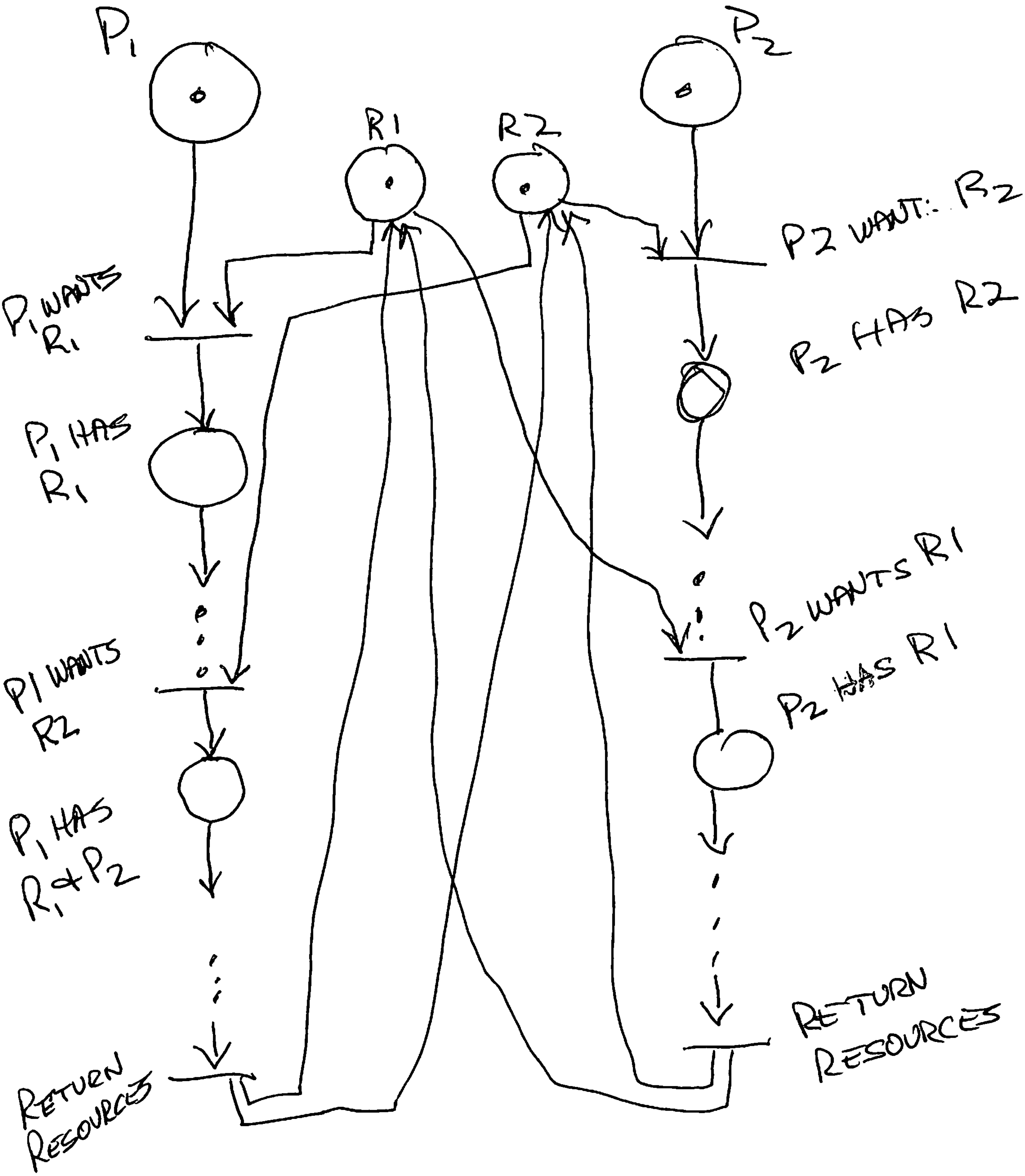
WEAKENS FOR FRS AND PETRI NETS

PETRI NET





PETRI NET (NONDET) AND CONCURRENCY



HOW HARD IS IT TO ANALYZE PETRI NETS?

TO DETERMINE IF SOME MARKING CAN EVENTUALLY ARISE IS IN

EXPSPACE(N)

SOLVABLE, BUT TAKES EXPONENTIAL SPACE

TIME IS ACTUALLY 2^{2^N}

IF PRIORITY ADDED TO TRANSITIONS, PETRI NETS ARE COMPLETE MODELS OF COMPUTATION.

$$\text{TOTAL} = \{ f \in \mathbb{N} \mid \forall x\ \varphi_f(x)\downarrow \}$$
$$= \{ f \in \mathbb{N} \mid W_f = \mathbb{N} \}$$

IS NOT RE.

TWO USEFUL SETS

TOTAL (ABOVE) IS NON-RE

$$\text{HALT} = \{ \langle f, x \rangle \mid f(x)\downarrow \}$$

IS RE, NON-RECURSIVE

INTRO TO REDUCTION

$A \leq_m B$ IF THERE EXISTS SOME COMPUTABLE ALGORITHM $f \ni$

$x \in A \iff f(x) \in B$

IF B IS EASY TO SOLVE THEN SO IS A IF f DOES NOT ADD TO COMPUTATIONAL COMPLEXITY

HOWEVER, IF A IS KNOWN TO BE HARD (OR EVEN UNSOLVABLE) AND f DOES NOT CHANGE THE COMPLEXITY LANDSCAPE, THEN B MUST BE HARD AT LEAST WITHIN THE ORDER OR f'S AND A'S COMPLEXITY, IF A IS UNSOLVABLE THEN SO IS B.

NOTIONS OF REDUCTIONS

$A \leq_1 B$ IFF $\exists$ AN ALG. f THAT IS 1-1

SUCH THAT $x \in A \Leftrightarrow f(x) \in B$

IF B IS SOLVABLE (SEMI-DECIDABLE)
THEN SO IS A, AND EACH ELEMENT
OF A HAS A UNIQUE COUNTERPART IN B,

$A \leq_m B$ IFF $\exists$ AN ALG. f THAT IS M-1
SUCH THAT $x \in A \Leftrightarrow f(x) \in B$
AGAIN B SOLVABLE (RE) THEN SO IS A
UNIQUE COUNTERPARTS ARE NOT REQUIRED
SO EACH $y \in B$ MAY HAVE MORE THAN
ONE (BUT ALSO MAYBE ZERO) ELEMENTS
FROM A THAT MAP TO IT

$\leq_1$ AND $\leq_m$ SAY NOTHING ABOUT
TIME OR SPACE COMPLEXITY OF f.

REDUCTION

HALT $\leq$ TOTAL

LET $\langle f, x \rangle$ BE ARB. PAIR OF NAT NUMBERS

DEFINE $f_x(y) = f(x)$ $\quad \forall y \in \mathbb{N}$

$\langle f, x \rangle \in$ HALT IFF $f(x)\downarrow$ $\quad$ REALLY φ_f AND φ_{f_x}

IFF $\forall y\ f_x(y)\downarrow$

IFF $f_x \in$ TOTAL

ANY ALGORITHM THAT SOLVES TOTAL

CAN BE USED TO SOLVE HALT,

SO HALT $\leq$ TOTAL

AND SINCE HALT IS UNDEC., SO IS TOTAL

CANNOT SHOW TOTAL $\leq$ HALT

AS TOTAL IS NOT EVEN RE,

BUT HALT IS

3/18 [illegible]

MORE REDUCTIONS

HALT $\leq$ ZERO $= \{f \mid \forall x\, \varphi_f(x) = 0\}$

LET $\langle f, x\rangle$ BE ARB. PAIR

DEFINE $\forall y\; f_x(y) = f(x) - f(x)$

$\langle f, x\rangle \in$ HALT IFF $f(x)\downarrow$ IFF $f(x) - f(x) = 0$

IFF $\forall y\, f_x(y) = 0$ IFF $f_x \in$ ZERO

SO ZERO IS UNDECIBLE, BUT IT'S WORSE

TOTAL $\leq$ ZERO

LET f BE ARBITRARY INDEX ($f \in \mathbb{N}$)

DEFINE $\forall x\; g(x) = f(x) - f(x)$

$f \in$ TOTAL IFF $\forall x\, f(x)\downarrow$

IFF $\forall x\, f(x) - f(x) = 0$

IFF $\forall x\, g(x) = 0$

IFF $g \in$ ZERO

ZERO

TOTAL

TOTAL $\leq$ ZERO

$$G_f(x) = f(x) - f(x)$$

$f \in$ TOTAL $\Leftrightarrow G_f \in$ ZERO

$$G_f(x) = f(x) - f(x) + x$$

$f \in$ TOTAL $\Leftrightarrow G_f \in$ IDENTITY

SO TOTAL $\leq$ IDENTITY

FORMALLY, WE SHOULD SAY

$$\varphi_f(x) - \varphi_f(x)$$

AND $\varphi_f(x) - \varphi_f(x) + x$

NOTION OF DEGREES OF UNSOLVABILITY

IF $A \leq_1 B$ AND $B \leq_1 A$ THEN

A AND B HAVE SAME DEGREE OF COMPLEXITY. IN FACT THEY ARE IN SAME 1-1 DEGREE AND HENCE REALLY ARE TIGHTLY COUPLED

IF $A \leq_m B$ AND $B \leq_m A$ THEN

A AND B ARE IN SAME M-1 DEGREE

FOR RE SETS IF C IS

1. RE
2. $A \leq_1 C$ FOR ALL RE SETS A

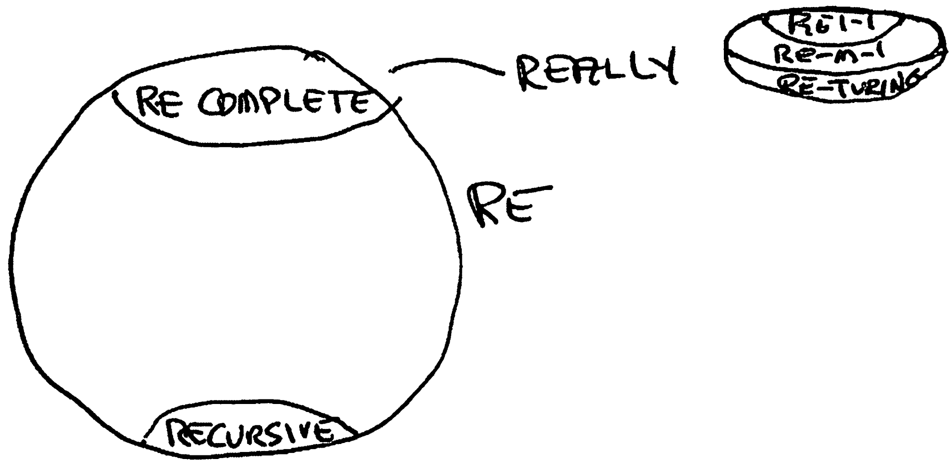
THEN C IS 1-1 COMPLETE

IF C IS

1. RE
2. $A \leq_m C$ FOR ALL RE SETS A

THEN C IS M-1 COMPLETE

RE- COMPLETE SETS



S IS RE-COMPLETE IFF

(a) S IS RE

(b) FOR ANY RE SET T, $T \leq S$

WE FOCUS ON $\leq_m$ OR EVEN $\leq_1$ FOR NOW

HALT $= K_0 = \{\langle f, x\rangle \mid \varphi_f(x)\downarrow\}$ IS RE-COMPLETE

(a) IT IS RE (CAN SEMI-DECIDE)

(b) LET T BE AN ARB RE SET

BY DEFINITION $\exists$ AN EFF PROC φ_t SUCH THAT $\mathrm{Dom}(\varphi_t) = T$, OR EQUIV. $\exists$ AN INDEX $t \ni T = W_t$ (ENUMERATION TH)

$x \in T$ IFF $x \in \mathrm{Dom}(\varphi_t)$

IFF $\varphi_t(x)\downarrow$

IFF $\langle t, x\rangle \in K_0 =$ HALT

SO $T \leq_1 K_0$

SINCE T IS ARB. RE, THIS SHOWS K_0 IS RE (1-1, m-1, TURING) COMPLETE.

$K = \{ f \mid \varphi_f(f)\downarrow \}$ IS RE COMPLETE

JUST SHOW $K_0 \leq K$

LET $\langle f, x \rangle$ BE ARB. PAIR ~~FROM~~ $\mathbb{N} \times \mathbb{N}$

DEFINE $\forall y\; f_x(y) = \varphi_f(x)$

LET INDEX OF f_x BE f_x (OVERLOAD)

$\langle f, x \rangle \in K_0$ IFF $x \in Dom(\varphi_f)$ IFF $\forall y\, [\varphi_{f_x}(y)\downarrow] \Rightarrow f_x \in K$

$\langle f, x \rangle \notin K_0$ IFF $x \notin Dom(\varphi_f)$ IFF $\forall y\, [\varphi_{f_x}(y)\uparrow] \Rightarrow f_x \notin K$

$K_0 \leq_1 K$, $K_0 \leq_m K$, $K_0 \leq_{TURING} K$

SO K_0 IS RE (1-1, M-1, TURING) COMPLETE

RICE'S THEOREM (STRONG) IN A PICTURE PART 1

HALT = $\{\langle x, y\rangle \mid \varphi_x(y)\downarrow\}$

LET P BE A NON-TRIVIAL PROPERTY OF PROCEDURES

$S_P \neq \emptyset$, $\overline{S_P} \neq \emptyset$ AS P NON-TRIVIAL

MOREOVER, IF f, g ARE PROCEDURE INDICES SUCH THAT $\forall x \; f(x) = g(x)$

NOTE! $\uparrow = \uparrow$

THEN EITHER BOTH f & g ARE IN S_P OR BOTH ARE IN $\overline{S_P}$

THIS MEANS f & g HAVE SAME I/O BEHAVIORS, ALTHOUGH THEY MAY HAVE RADICALLY DIFFERENT IMPLEMENTATIONS

RICE'S THEOREM (STRONG) IN A PICTURE PART 2

AGAIN P NON-TRIVIAL SO

1. $\exists r \ni r \in S_P = \{f \mid \varphi_f$ HAS PROPERTY P

AGAIN P CARES ONLY ABOUT I/O BEHAVIOR

2. ALL INDICES IN
   " $EMPTY = \{f \mid \forall x \; \varphi_f(x)\uparrow\}$
   ARE EITHER IN S_P OR $\overline{S_P}$

WLOG, ASSUME IF $f \in EMPTY$ THEN $f \notin S_P$
IF NOT SO, WE COMPLEMENT P AND IT IS
SO, AS P IS DECIDABLE IFF NOT P IS DEC.

LET x, y BE ARBITRARY
DEFINE, USING $r \in S_P$

$$F_{r,x,y}(z) = \varphi_x(y) - \varphi_x(y) + \varphi_r(z)$$

RICE'S THEOREM (STRONG) IN A PICTURE PART 3

LET x, y BE ARBITRARY

THEN $\langle x, y \rangle \in$ HALT $\Leftrightarrow \varphi_x(y)\downarrow$

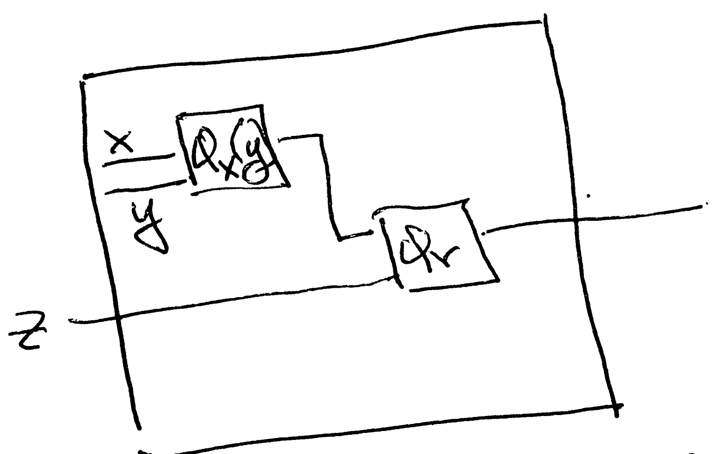
LET P BE NON-TRIVIAL I/O PROPERTY

(a) $r \in S_P$ IS SOME ELEMENT IN S_P

SO φ_r HAS PROPERTY P

(b) WLOG $\emptyset \in \overline{S_P}$. REALLY, IF $\mathrm{Dom}(\varphi_t) = \emptyset$ THEN $t \in \overline{S_P}$

DEFINE $F_{r,x,y}$ BY



IS φ_r IF $\varphi_x(y)\downarrow$

IS $\emptyset$ IF $\varphi_x(y)\uparrow$

$$F_{r,x,y}(z) = \varphi_x(y) - \varphi_x(y) + \varphi_r(y)$$

$F_{r,x,y} \in S_P \Leftrightarrow \langle x, y \rangle \in$ HALT

SO HALT $\leq_1 S_P$ AND P IS UNDECIDABLE

APPLYING RICE'S THEOREM

$\text{HASZERO} = \{ f \mid \exists x\ f(x) = 0 \}$

IS UNDECIDABLE BY RICE'S THEOREM

1. HASZERO IS NON-TRIVIAL

   $C_0(x) = 0 \in \text{HASZERO}$

   $S(x) = x+1 \notin \text{HASZERO}$

2. HASZERO IS IMMUNE TO IMPLEMENTATION

   LET f, g BE SUCH THAT $\forall x\ f(x) = g(x)$

   $f \in \text{HASZERO} \iff \exists x\ f(x) = 0$

   LET x_0 BE SOME SUCH VALUE, $f(x_0) = 0$

   BUT THEN $g(x_0) = 0$ AND SO

   $\implies \exists x\ g(x) = 0 \implies g \in \text{HASZERO}$

   $* f \notin \text{HASZERO} \iff \forall x\ f(x) \neq 0$

   $\iff \forall x\ g(x) \neq 0$

   AS $\forall x\ g(x) = f(x)$

   $\iff g \notin \text{HASZERO}$

APPLYING RICE'S THEOREM

$MI = \{ f \mid \forall x\ f(x) < f(x+1) \}$

IS UNDECIDABLE BY RICE'S THEOREM

1. MI IS NON-TRIVIAL

   $S(x) = x+1 \in MI$

   $C_0(x) = 0 \notin MI$

2. MI IS IMMUNE TO IMPLEMENTATION

   LET f, g BE SUCH THAT $\forall x\ f(x) = g(x)$

   $f \in MI \Leftrightarrow \forall x\ f(x) < f(x+1)$

   $\Leftrightarrow \forall x\ g(x) < g(x+1)$

   AS $\forall x\ g(x) = f(x)$

   $\Leftrightarrow g \in MI$

NOTE 1: WE REALLY TALK ABOUT φ_f + φ_g

BUT OVERLOADING IS FINE

NOTE 2: MI = MONOTONICALLY INCREASING

NOTES ON REDUCTION

$\text{TOTAL} = \{ f \mid \forall x\ f(x)\downarrow \}$

$\text{TOTAL} \leq_1 \text{MI} = \{ f \mid \forall x\ f(x) < f(x+1) \}$

LET f BE ARBITRARY AND DEFINE

$\forall x\ G_f(x) = f(x) - f(x) + x$

$$\begin{aligned} f \in \text{TOTAL} &\Rightarrow G_f(x) = x \\ &\Rightarrow G_f \in \text{MI} \end{aligned}$$

$$\begin{aligned} f \notin \text{TOTAL} &\Leftrightarrow \exists x\ f(x)\uparrow \\ &\Rightarrow G_f(x)\uparrow \text{ FOR SOME } x \\ &\Rightarrow G_f \notin \text{MI} \end{aligned}$$

THUS, $\text{TOTAL} \leq \text{MI}$

AND MI IS NOT RE

THAT IS, NOT SEMI-DECIDABLE

THIS IS A STRONGER RESULT
THEN RICE'S CAN GET US.