Tutoraggio

Linguaggi regolari

$L \in$ Regolare $\iff$ DFA

1.

$val : \Sigma^* \to \mathbb{N}$

$\Sigma = \{0,1\}$ $\quad L = \{\sigma \in \Sigma^* \mid val(\sigma) \equiv 0 \pmod 3\}$

(numeri divisibili per 3)

Esempi $11 = 3 \in L$

$n \% 3$

$0 \to 0$	$3 \to 0$	$6 \to 0$
$1 \to 1$	$4 \to 1$	$7 \to 1$
$2 \to 2$	$5 \to 2$	$8 \to 2$

↳ classi di equivalenza

Aggiungere 0 $\to$ $\underbrace{\underbrace{0101}_{5}\underline{0}}_{10}$ $\to \cdot 2$

Aggiungere 1 $\to$ $\underbrace{\underbrace{0101}_{5}\underline{1}}_{11}$ $\to \cdot 2 + 1$

Classi di equivalenza

$n = 3K + 0$ Resto 0

$n = 3K + 1$ Resto 1

$n = 3K + 2$ Resto 2

$n = 3K + 0$

Aggiungere 0

$n = 3K \to 2n = 3K$

$n = 3\frac{K}{2} \underline{+0}$

Aggiungere 1

$n = 3K \to 2n + 1 = 3K$

$2n = 3K - 1$

$-1 \to 2$
$0 \to 0$
$1 \to 1$
$2 \to 2$
$3 \to 0$

$2n = 3K + 2$

$n = 3\frac{K}{2} \underline{+1}$

$n = 3K + 1$

Aggiungere 0

$2n = 3K + 1$

$2n = 3K - 2$

Aggiungere 1

$2n \cancel{+1} = 3K \cancel{+1}$

$n = 3\frac{K}{2} \underline{+0}$

$n = 3k + 2$

Aggiungere 0

$2n = 3k + 2$

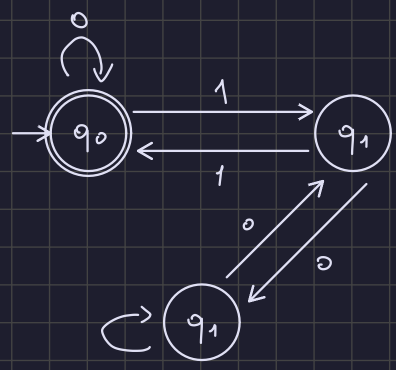
$n = 3\frac{k}{2} + 1$

Aggiungere 1

$2n + 1 = 3k + 2$

$2n = 3k + 1$

$n = 3\frac{k}{2} + 1$



Dimostrazione

$x \in L \iff \hat{\delta}(q_0, x) \in F$

Separando il se e solo se bisogna dimostrare le seguenti implicazioni:

1) $x \in L \Rightarrow \hat{\delta}(q_0, x) \in F$
2) $x \in L \Leftarrow \hat{\delta}(q_0, x) \in F \quad \equiv \quad x \notin L \Rightarrow \hat{\delta}(q_0, x) \notin F$

Dimostro per induzione sulla lunghezza della stringa : $|\sigma|$

Caso base

È una stringa lunga il minimo possibile per mostrare sia che appartiene sia che non appartiene

$|\sigma| = 1$

$\sigma = 0 \Rightarrow \hat{\delta}(q_0, 0) = q_0 \in F$

$\sigma = 1 \Rightarrow \hat{\delta}(q_0, 1) = q_1 \notin F$

Non bisogna necessariamente coprire tutti gli stati dell'automa nei casi base perchè basta una stringa che finisce in uno stato finale e una che non finisce in uno stato finale

Passo induttivo

Ipotesi induttiva

1) $\forall \sigma . \exists n > 1 . |\sigma| \leq n : \sigma \in L \Rightarrow \hat{\delta}(q_0, \sigma) = \{q_0\} \in F$
2) $\forall \sigma . \exists n > 1 . |\sigma| \leq n : \sigma \notin L \Rightarrow \hat{\delta}(q_0, \sigma) = \{q_1, q_2\} \notin F$

Prendo una stringa di lunghezza n+1:

Sigma è una stringa binaria multiplo di 3

$\sigma = \sigma' \alpha \quad . \quad |\sigma'| = n, \alpha \in \Sigma$

$- \sigma \in L, |\sigma| = n + 1$

- $\sigma = \sigma' 0 \rightarrow \quad \hat{\delta}(q_0, \sigma)$

$$= \hat{\delta}(q_0, \sigma' 0)$$

$$= \delta(\hat{\delta}(q_0, \sigma'), 0)$$

$$\downarrow \text{i.i}$$

$$= \delta(q_0, 0) = q_0 \in F$$

- $\sigma = \sigma' 1 \rightarrow \quad \hat{\delta}(q_0, \sigma)$

$$= \hat{\delta}(q_0, \sigma' 1)$$

$$= \delta(\hat{\delta}(q_0, \sigma'), 1)$$

$$\downarrow \text{i.i}$$

$$= \delta(q_1, 1) = q_0 \in F$$

- $\sigma \notin L, \ |\sigma| = n+1$
  - $\sigma = \sigma' 0 \rightarrow \hat{\delta}(q_0, \sigma)$

$$= \hat{\delta}(q_0, \sigma' 0)$$

$$= \delta(\hat{\delta}(q_0, \sigma'), 0)$$

$$\downarrow \text{ii.}$$

$$= \delta(\{q_1, q_2\}, 0) = \{q_2, q_1\} \notin F$$

  - $\sigma = \sigma' 1 \rightarrow \hat{\delta}(q_0, \sigma)$

$$= \hat{\delta}(q_0, \sigma' 1)$$

$$= \delta(\hat{\delta}(q_0, \sigma'), 1)$$

$$* \xrightarrow{\text{ii 1}} = \delta(q_0, 0) = q_1 \notin F$$

$$* \xrightarrow{\text{ii 2}} = \delta(q_2, 0) = q_2 \notin F \qquad \square$$

Pumping lemma linguaggi regolari

$L \in Reg \Rightarrow$ Pumping lemma

$$\Downarrow$$

$$z \in L,\ |z| \geq n,\ \exists\, z = uvw,\ |uv| \leq n,\ |v| > 0,\ \forall i \in \mathbb{N}.\ uv^iw \in L$$

$\neg$ Pumping lemma $\Rightarrow L \notin Reg$

Definizione

Sia $L \in REG$, allora:

$$\exists n \in \mathbb{N}.\ \forall z \in L.\ |z| \geq n \Rightarrow \exists\, u, v, w \in \Sigma^*.\ \underbrace{z = uvw \wedge \begin{cases} |uv| \leq n \\ |v| > 0 \\ \forall i \geq 0.\ uv^iw \in L \end{cases}}$$

$$(|z| < n \vee \exists u, v, w \in \Sigma^*)$$

$$a \rightarrow b \equiv \neg a \vee b$$

$$\Downarrow \equiv$$

$$\Big(\underbrace{\neg(z = uvw) \vee \neg(|uv| \leq n) \vee \neg(|v| > 0)}_{\neg\alpha} \vee \underbrace{\neg(\forall i \geq 0.\ uv^iw \in L)}_{\beta}\Big)$$

Negazione

$$\forall n \in \mathbb{N}.\ \exists z \in L.\ |z| < n \wedge \forall u, v, w \in \Sigma^*.$$

$$.\ \big(z = uvw \wedge |uv \leq n \wedge |v| > 0\big) \Rightarrow \exists i \geq 0.\ uv^iw \notin L$$

1. $L = \{ 0^n 1^m \mid n \neq m \}$

Condizioni di appartenenza

$|0| \neq |1|$

Consideriamo la stringa

$z = 0^k 1^{k+1}$ $\quad n = k$, $m = k+1$

Pompiamo la stringa

$z_i = 0^{k+(i-1)|v|} 1^{k+1}$

- $i = 0$

$z_0 = 0^{k-|v|} 1^{k+1}$

$|0| \neq |1| \Rightarrow k - |v| \neq k+1 \Rightarrow |v| \neq -1$ ✓

- $i = 1$

$z_1 = 0^{k+|v|} 1^{k+1}$

$|0| \neq |1| \Rightarrow k + |v| \neq k+1 \Rightarrow |v| \neq 1$ ✓

Non si riesce a trovare un pompaggio che rompe le condizioni del pumping lemma utilizzando la stringa z.
Bisogna quindi usare un altro approccio.

Per le proprietà di chiusura sappiamo che: $L \in Reg \Leftrightarrow \overline{L} \in Reg$

Quindi se dimostriamo che complemento di L non è regolare con il pumping lemma negato, allora anche L originale non è regolare

$\overline{L} = \{ 0^n 1^m \mid n = m \}$

Scegliamo una stringa

$z = 0^k 1^k$ $\quad n = m = k$

Pompiamo la stringa

$$z_i = 0^{k+(i-1)|v|} 1^k$$

- $i = 2$

$$z_2 = 0^{k+|v|} 1^k$$

$$|0| = |1| \Rightarrow k+|v| = k \Rightarrow |v| = 0$$ Viola i vincoli del pumping lemma

Abbiamo quindi dimostrato che $\bar{L}$ non è regolare e quindi non è regolare nemmeno L

2. $$L_m = \{ x2y2x^R \mid y \in \{0\}^*, |y| \leq m, x = 1^{n^m} \} \quad m \in \mathbb{N}$$

Siccome x è formata solo da 1 il reverse di x non ha alcun effetto

- $$L_0 = \{ x2y2x^R \mid y \in \{0\}^*, |y| \leq 0, x = 1^{n^0} = 1 \}$$
  $$= \{1221\}$$ Qualunque linguaggio FINITO è regolare $\Rightarrow L_0 \in Reg$

- $$L_1 = \{ x2y2x^R \mid y \in \{0\}^*, |y| \leq 1, x = 1^n \}$$
  $$= \{1^n 2 \quad 2 1^n\} \cup \{1^n 2 0 2 1^n\}$$

  Sembra non sia regolare perchè ci sono due "gruppi" vincolati tra di loro (1^n) cioè il numero di uni deve essere uguale sia a sinistra che a destra. Dimostriamo quindi che il linguaggio non è regolare:

  Prendiamo una stringa z

  $$z = 1^k 2 2 1^k \in L \quad n = k$$

  $u\ v$ ($\leq k$), w

  Pompiamo la stringa

  $$z_i = 1^{k+(i-1)|v|} 2 2 1^k$$

  - $i = 2 \quad z_2 = 1^{k+|v|} 2 2 1^k$

  $$|1|_{sx} = |1|_{dx} \Rightarrow k+|v| = k \Rightarrow |v| = 0$$ Viola i vincoli del pumping lemma

  Quindi il linguaggio L_1 non è regolare

- $$L_2 = \{ x2y2x^R \mid y \in \{0\}^*, |y| \leq 2, x = 1^{n^2} \}$$

  Questo linguaggio non è nè regolare e nè CF, quindi da m ≥ 2 si ottengono soltanto linguaggi non CF (da dimostrare con il pumpuing lemma dei CF)

2.1 L'unione è regolare?

Verifichiamo l'unione di tutti i linguaggi è regolare

$$\bigcup_m L_m = L'$$

$$= \{ x 2 0^m 2 x^R \mid x = 1^{n^m} \}$$ (In un unione è come se la variabile "sparisse")

$$= \{ 1^{n^m} 2 0^m 2 1^{n^m} \mid x = 1^{n^m} \}$$

Questo linguaggio non è CF (da dimostrare con il pumpuing lemma dei CF)

L'intersezione è regolare?

Verifichiamo se l'intersezione di tutti i linguaggi è regolare

$$\bigcap_m L_m = L'$$

$$= L_0 \cap L_1 \cap L_2 \cap \dots \cap L_m$$

$$= \{1221\} \cap \{1221, \dots\} \cap \dots \cap L_m$$

$$= \{1221\}$$

Siccome il linguaggio è finito, allora è regolare

3. $L = \{ 0^m 0^{2n} 0^{k+m} \mid m, n, k > 0 \}$

$$= \{ 0^{2m+2n+k} \mid m, n, k > 0 \}$$

Siccome le variabili sono su un solo simbolo, e quindi non ci sono dipendenze tra simboli diversi, si può dire che il linguaggio è regolare.

4. $L_p = \{ a^{p \cdot m} b^m a^{p \cdot n} \mid m, n \geq 1 \} \quad p > 0$

- $L_1 = \{ a^m b^m a^n \mid m, n \geq 1 \}$ Sembra CF perchè a e b dipendono entrambi dalla stessa variabile m

  Dimostriamo che non è regolare con il pumping lemma

  Prendiamo una stringa nel linguaggio

  $z = a^k b^k a \in L$

  Pompiamo la stringa

  $z_i = a^{k + (i-1)|v|} b^k a$

  - $i = 0$

    $z_0 = a^{k-|v|} b^k a$

    $|a| = |b| \Rightarrow k - |v| = k \Rightarrow |v| = 0$ Viola i vincoli del pumping lemma $\quad z_i \notin L_1$

- $L_2 = \{ a^{2m} b^m a^{2n} \mid m, n \geq 2 \}$ CF

  $z_i = a^{2k} b^k a^2 \in L$

$$z_i = a^{2k+(i-1)|v|} b^k a^2$$

- $i=0$

$$z_0 = a^{2k-|v|} b^k a^2$$

$$|a| = 2|b| \Rightarrow 2k-|v| = 2k \Rightarrow |v|=0$$ Viola i vincoli del pumping lemma $\quad z_i \notin L_2$

Osserviamo che con p≥1 i linguaggi sono tutti CF

4.1 L'unione è regolare?

Verifichiamo l'unione di tutti i linguaggi è regolare

$$\bigcup_p L_p = L'$$
$$= \{a^{pm} b^m a^{pn} \mid m, n, p \geq 1\}$$

Sicuramente non è regolare. Possiamo vedere che non è CF perchè a è legato a b per la variabile m, e la a a sinistra è legata a quella a destra per la variabile p.

$$= \{a^{pm} b^m c^{pn} \mid m, n, p \geq 1,\ c = a\}$$

Le condizioni di appartenenza sono:

$$\begin{cases} 1)\ |a| = p|b| \\ 2)\ |a|n = |c|m \end{cases}$$

Pumping lemma:

Prendiamo una stringa nel linguaggio

$$z = a^{k^2} b^k c^{k^2} \in L \quad m=n=p=k$$

Pompiamo la stringa

$$z_i = a^{k^2+(i-1)|v|} b^k c^{k^2}$$

- $i=2$

$$z_2 = a^{k^2+|v|} b^k c^{k^2}$$

1) $|a| = p|b| \Rightarrow k^2+|v| = k \cdot k \Rightarrow |v|=0$ Viola i vincoli del pumping lemma

2) $|a|n = |c|m \Rightarrow (k^2+|v|)k = k^2 \cdot k \Rightarrow |v|=0$ Viola i vincoli del pumping lemma

$\Downarrow$

$z_i \notin L$

L'unione dei linguaggi non è regolare

L'intersezione è regolare?

Verificiamo l'intersezione di tutti i linguaggi è regolare

$$\bigcap_p L_p = L'$$

$$= L_1 \cap L_2 \cap \dots \cap L_p$$

$$= \{a^m b^m a^n, m,n \geq 1\} \cap \{a^{2m} b^m a^{2n}, m,n \geq 1\} \cap \dots \cap L_p$$

$$= \emptyset \in Reg$$

Linguaggi Context Free

$x \in L \iff x \in L(G)$, $\quad G = \langle V, T, P, S \rangle$

$S \in V$

$P: X \in V \to \alpha \in (V \cup T)^*$

Se $P = \emptyset \to G = \langle \{S, x, y, \dots\}, \{a, b, c, \dots\}, \emptyset, S \rangle \to L(G) = \emptyset$

Un linguaggio può essere generato da più grammatiche

1. $L = \{a^n b^n \mid n \neq m\} \notin Reg$

Bisogna trovare una grammatica e dimosrtrarla induttivamente

$$G = \begin{cases} S \to aSb \mid A \mid B \\ A \to aA \mid a \\ B \to bB \mid b \end{cases}$$

oppure

$n \neq m \equiv \underbrace{n > m}_{G_1} \lor \underbrace{n < m}_{G_2}$

$G' = G_1 \cup G_2$

$G_1 = A \to aAb \mid aA \mid a$

$G_2 = B \to aBb \mid b \mid Bb$

$$G' = \begin{cases} S \to A \mid B \\ A \to aAb \mid aA \mid a \\ B \to aBb \mid b \mid Bb \end{cases} \qquad (G_1 \lor G_2)$$

Dimostrazione di G'

Tesi:

1) $x \in L \Rightarrow S \to^* x$

2) $S \to^n x \Rightarrow x \in L$

1) $x' \in L \Rightarrow S \to^* x'$

$|a| > |b|$:

Caso base

$|x| = 1$

$x = a \Rightarrow S \to A \to a$

$x = b \Rightarrow S \to B \to b$

Passo induttivo

Ipotesi induttiva $x = a^{n+h} b^n$, $h > 0 \Rightarrow S \to^* x \equiv S \to A \to^* x$

Consideriamo una stringa x' lunga n+1 $|x'| > |x|$

1. $x' = a^{h+(n+1)} b^{n+1}$

$= a \underbrace{a^{n+h} b^n}_{x} b \implies S \to A \to aAb \xrightarrow{ii}{}^{*} axb \equiv x'$

2. $x' = a^{n+(h+1)} b^n$

$= a\, a^{n+h} b^n \implies S \to A \to aA \xrightarrow{ii} ax \equiv x'$

$|a| < |b|$ analogo

2) $S \to^n x \implies x \in L$

$|a| > |b|$:

Caso base

$n = 2$ (numero minimo di passi)

$S \to A \to a$

$S \to B \to b$

Passo induttivo

Ipotesi induttiva: $S \to^n x \equiv S \to A \to^{n-1} x$

- $\underbrace{S \to A \to aA \xrightarrow{ii}{}^{n-1} ax}_{n+1 \text{ passi}} \in L$

- $\underbrace{S \to A \to aAb \to^{n-1} axb}_{n+1 \text{ passi}} \in L$

$|a| < |b|$: analogo

2. $L = \{ \sigma c^n d^m \mid \sigma \in \{a,b\}^*, \ |\sigma| = |c|, \ n, m \in \mathbb{N}$

È intuitivamente context free perchè c'è un legame tra il gruppo sigma e il gruppo c, mentre invece il gruppo d è indipendente dagli altri

$S \to Sd \mid D$

$D \to aDc \mid bDc \mid \varepsilon$

Questo linguaggio è formato da due gruppi indipendenti quindi si possono separare e unire due grammatiche

$S \to AD$

$D \to dD \mid \varepsilon$

$A \to aAc \mid bAc \mid \varepsilon$

3\. $L = \{ a^n b^m c^n d^m \mid n, m > 0 \}$

Questo linguaggio non è CF (Dimostrare con PL dei CF)

4\. $L = \{ \sigma \in \Sigma^* \mid \Sigma = \{0, 1\}, |\sigma|_0 = |1|_1^2 \}$

I linguaggi CF possono generare soltanto linguaggi lineari, quindi questo linguaggio non è CF

Pumping lemma context free

Sia $L \in CF$, allora:

$$\exists n \in \mathbb{N} .\ \forall z \in L .\ |z| \geq n \Rightarrow \exists u, v, w, x, y \in T^* .\ z = uvxy \land \begin{cases} |vwx| \leq n \\ |vx| > 0 \\ \forall i \geq 0 .\ uv^iwx^iy \in L \end{cases}$$

Negazione

$$\forall n \in \mathbb{N} .\ \exists z \in L .\ |z| \geq n \Rightarrow \forall u, v, w, x, y \in T^* .$$
$$.(z = uvwxy \land |vwx| \leq n \land |vx| > 0) \Rightarrow \exists i \geq 0 .\ uv^iwx^iy \notin L$$

Esempio: $L = \{0^n 1^{2^n} \mid n \geq 0\}$

Non è intuitivamente regolare perchè non si riesce a creare un automa che lo riconosce.
Non è nemmeno intuitivamente context free perchè contiene una funzione non lineare.

$$z = 0^n 1^{2^n} \rightarrow |z| = n + 2^n \geq n$$

Esistono diverse combinazioni di suddivisioni e si devono verificare tutte:

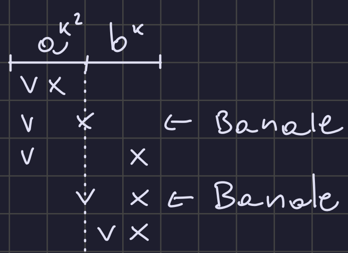
Questi due casi sono spesso banali perchè rompono lo schema del linguaggio

1. $L = \{a^{m \cdot n} b^n \mid n, m \geq 1\}$

Scegliamo una stringa z

- $K = m = n \geq 1$

$$z = a^{K^2} b^K$$



1. $(vx) \in a^{K^2}$

$$z' = a^{K^2 + (i-1)|vx|} b^K \qquad |a| = |b| \cdot l \quad . \quad l \geq 1$$

- $i = 2$

$$K^2 + |vx| = K \cdot l$$

$$K^2 - Kl + |vx| = 0$$

$$K = \frac{l \pm \sqrt{l^2 - 4|vx|}}{2}$$

Se |vx| = 1 la stringa appartiene ancora al linguaggio,
quindi con questo pompaggio non si riesce ad uscire dal linguaggio.

- $m = 1 \quad n = K \rightarrow |a| = |b| \cdot l \quad . \quad l \geq 1$

$$z = a^K b^K$$

1) $z' = a^{K + (i-1)|vx|} b^K$

- $i=2$

$z'_2 = a^{k+|vx|} b^k$

$k+|vx| = kl$

$k-kl = |vx|$

$k((l-1) = |vx|$

$\left.\begin{array}{l} l=1 \rightarrow |vx|=0 \\ l>1 \rightarrow |vx|>k \end{array}\right\}$ Assurdo $(|vx|>0 \wedge |vwx| \leq k)$

2) $z' = a^{k+(i-1)|v|} b^{k+(i-1)|x|}$

- $i=2$

$z'_2 = a^{k+|v|} b^{k+|x|}$

$k+|v| = (k+|x|)l$

$k+|v| = kl+|x|l$

$k((l-1) = |v|-|x|l$

$l=1 \rightarrow |v|=|x|$ Non si può dire niente a riguardo

Quindi questo pompaggio non fa uscire la stringa dal linguaggio, si deve trovare un'altra stringa

- $m=k \quad n=1 \rightarrow |a| = l \,.\, l \geq 1$

$z = a^m b$

1. $z' = a^{m+(i-1)|vx|} b$

- $i=2$

$k+|vx| = l$ Non si può dire nulla a riguardo

Questo pompaggio non porta la stringa fuori dal linguaggio, quindi bisogna trovarne un'altra

- $m=k \quad n=p \,.\, p$ é primo $>k \rightarrow |a| = |b|l$

$z = a^{kp} b^p$

1. $z' = a^{kp+(i-1)|vx|} b^p$

- $i=2$

$kp+|vx| = pl$

$p(l-k) = |vx|$

Se $k < l \rightarrow p(l-k) = |vx| = hp \geq p > k$

Se $k = l \rightarrow p(l-k) = |vx| = 0$

Se $k > l \rightarrow p(l-k) = |vx| < 0$

Assurdo

2. $z' = a^{kp+(i-1)|v|}\, b^{p+(i-1)|x|}$

- $i=2$

$z'_2 = a^{kp+|v|}\, b^{p+|x|}$

$kp+|v| = pl+|x|l$

$kp - pl = |x|l - |v|$

$p(k-l) = |x|l - |v|$

se $l < k \rightarrow p = |x|(k-1) - |v|$

Se $k=1 \rightarrow |v| = -p$ Assurdo

se $k>1 \rightarrow p = |x| - |v|$

$|x| = p + |v| > k$ Assurdo

Se $l = k \rightarrow |x|k - |v| = 0$

$|v| = k|x|$ Assurdo perché se $k=1$ e $|v|=1 \rightarrow |vx| = 2 \rightarrow |vx| < k$ $2 < 1$ ✗

Se $l > k \rightarrow -p = |x|l - |v|$

$|x|l = |v| - p$

Se $l = p \rightarrow |x|p = |v| - p$

$p(|x|+1) = |v|$ Assurdo $\forall p$

Se $l = k+1 \rightarrow |x|k + |x| = |v| - p$

Se $k+1 = p$

$|x|p + |x| = |v| - p$ Assurdo $\forall p$

3. $z' = a^{kp}\, b^{p+(i-1)|vx|}$

- $i=2$

$z'_2 = a^{kp}\, b^{p+|vx|}$

$kp = pl + |vx|l$

$p(k-l) = |vx|l$

Se $l < k \rightarrow p = |vx|(k-1)$

Se $k = 1 \rightarrow |vx| = p$ Assurdo $\forall k$

Se $l > k \rightarrow -p = |vx|(k+1)$ Assurdo $\forall k$

Abbiamo dimostrato che tutte le possibili suddivisioni portano all'assurdo, quindi il linguaggio non è context free.

Ulteriore soluzione: $m = k$ $n = k^2$

Insiemi ricorsivi

1.

$A = \{ x \mid \exists y \in \mathbb{N} \,.\, x = y^2 + 1 \}$

Definiamo la funzione caratteristica

$$f_A(x) = \begin{cases} 1 & x \in A \\ 0 & x \notin A \end{cases}$$

Bisogna fornire lo pseudocodice che calcola la funzione caratteristica:

```
input(x)
if x ≤ 1 return x;
for y = 1 to x {
  if x == y^2 + 1 {
    return 1;
  }
}
return 0;
```

2.

$A = \{ x \mid \varphi_x(x^2)$ non converge in meno di n passi $\}$

Definiamo la funzione caratteristica

$$f_A(x) = \begin{cases} 1 & x \in A = \varphi_x(x)^2 \not\downarrow \text{ in meno di n passi} \\ 0 & x \notin A = \varphi_x(x)^2 \downarrow \text{ in meno di n passi} \end{cases}$$

Bisogna fornire lo pseudocodice che calcola la funzione caratteristica:

```
input(x,n)
costruisco phi_x  // Esiste una procedura algoritmica che la costruisce

y = x^2
for i = 1 to n {
  esegui uno step di phi_x(y)
  if phi_x(y) ha terminato {
    return 0
  }
}
return 1
```

3.

$A = \{ x \mid \exists y \in \mathbb{N} \,.\, x = y^3 + y^2 + y \}$

Definiamo la funzione caratteristica

$$f_A(x) = \begin{cases} 1 & x \in A \\ 0 & x \notin A \end{cases}$$

Bisogna fornire lo pseudocodice che calcola la funzione caratteristica:

```
input(x)
for y = 0 to x {
  z = y^3 + y^2 + y
  if x == z {
    return 1;
  }
}
return 0;
```

4.

$A = \{ x^2 \mid \varphi_x(x^3+1)$ non converge in 2 passi $\}$

Definiamo la funzione caratteristica

$$f_A(x) = \begin{cases} 1 & x \in A \\ 0 & x \notin A \end{cases}$$

Bisogna fornire lo pseudocodice che calcola la funzione caratteristica:

```
input(x)
if x non è un quadrato return 0

y = sqrt(x)
costruisco phi_y # Esiste un procedimento algoritmico che la calcola

z = y^3 + 1
for i = 1 to 2 {
  eseguo il prossimo step di phi_y(z)
  if phi_y(z) converge then return 0
}

return 1
```

Insiemi creativi

$A \in \text{Creativi} \Rightarrow \bar{A} \in \text{Produttivi}$

Bisogna dimostrare:
1) A è ricorsivamente enumerabile

2) A è creativo

1) $A \in RE \quad \land \quad \bar{K} \leq_F \bar{A} \iff K \leq_F A$

2) $K \leq_F A \iff \bar{K} \leq_F A \quad \land \quad \bar{K} \leq_F \bar{A} \iff K \leq_F A$

L'insieme K è l'insieme di elementi che terminao su quel programma.

$$K = \{x \mid \varphi_x(x)\} = \{x \mid x \in W_x\}$$

Nei produttivi il programma non riesce a dare una risposta.
Nei creativi il programma da una risposta solo quando la trova.

1. $A = \{x \mid W_x = 2\mathbb{N}\}$

Se un insieme è finito sarà sempre ricorsivo, in questo caso l'insieme è infinito e rappresenta tutti i programmi che terminano SOLO con un numero pari in input.

$$\bar{A} = \{x \mid W_x \neq 2\mathbb{N}\}$$

Per verificare che l'input sia pari (o dispari) c'è bisogno di un insieme infinito di passi. Questa è la definizione degli insiemi produttivi, si ha sempre un input che sfugge all'enumerazione. Nell'insieme A sappiamo che i numeri dispari NON sono compresi nell'insieme, quindi i numeri dispari devono far divergere, ma non si può trovare la non terminazione, quindi entrambi gli insiemi sono produttivi.

2. $A = \{x \mid W_x = \emptyset\}$

Siccome l'insieme è vuoto non si può convergere per nessun input, quindi è produttivo.

```
while j = 0 {
  exec phi_x(j)
  j++
}
```

Questo while non funziona perchè se l'input j diverge non si arriva mai all'istruzione j++.

$$\bar{A} = \{x \mid W_x \neq \emptyset\}$$

Siccome l'insieme è diverso dal vuoto, almeno uno ci deve essere, quindi con un dovetail si può trovare eventualmente un input che fa convergere. Quindi è creativo.

Bisogna quindi dare un doppio ciclo (dovetail):

```
y = 0
while true {
  for j = 0 to y {
    esegui un passo di phi_x(j)
    j++
  }
  y++
}
```

3. $A = \{x \mid W_x \subseteq 2\mathbb{N}\}$

È produttivo perchè il suo complemento è creativo

$\bar{A} = \{x \mid W_x \not\subseteq 2\mathbb{N}\}$

È creativo perchè basta trovare UN elemento che non appartiene a 2N.

4. $A = \{x \mid W_x \subseteq \mathbb{N}\}$

La condizione è sempre vera

$\bar{A} = \{x \mid W_x \not\subseteq \mathbb{N}\}$

La condizione non è mai vera

Siccome questi insiemi sono entrambi ricorsivi e possono essere riconosciuti da un automa, quindi sono anche regolari. C'è un automa che riconosce tutto e uno che non riconosce niente.

5. $A = \{x \mid W_x \overset{=}{\subseteq} \emptyset\}$ Produttivo. Sottoinsieme uguale di vuoto equivale a dire uguale al vuoto

$\bar{A} = \{x \mid W_x \underset{\neq}{\not\subseteq} \emptyset\}$ Creativo. Diverso da sottoinsieme uguale di vuoto equivale a dire diverso dal vuoto.

6. $A = \{x \mid [0, m] \subseteq W_x\}$ Creativo

Bisogna testare un numero finito di input e verificare se termina su tutti.

$\bar{A} = \{x \mid [0, m] \not\subseteq W_x\}$ Produttivo

Deve esserci almeno un elemento dell'intervallo per cui diverge, però non si può decidere la non terminazione.

7. $A = \{x \mid 2\mathbb{N}+1 \subseteq W_x\}$

Produttivo perchè bisogna controllare tutti i numeri dispari

$\bar{A} = \{x \mid 2\mathbb{N}+1 \not\subseteq W_x\}$

Produttivo perchè diverge solo sui pari e non si può decidere la non terminazione.

8. $A_m = \{x \mid [0, m] \subseteq W_x\}$

Intuitivamente l'insieme è creativo.

Dimostro

1) $A \in RE$

$$\psi(x) = \begin{cases} 1 & se\ x \in A \\ \uparrow & altrimenti \end{cases}$$ non si può decidere la terminazione

```
input(x)
l[m] = [] // Inizializzo una lista vuota
costruisco phi_x(y) // Esiste una procedura che lo fa
while true {
  for y = 0 to m {
    esegui un passo di phi_x(y)
    if phi_x(y) ha terminato {
      l[y] = 1
    }
  }

  if tutti i valori di l == 1 {
    return 1
  }
}
```

oppure

```
input(x)

for y = 0 to m {
  exec phi_x(y)
}
return 1
```

(Se tutti i programmi terminano si esce dal for e si ritorna 1. Se almeno un programma non termina allora il programma diverge)

2) $A \in CR$:

$K \leq_F A \iff \bar{K} \leq_F \bar{A}$

$\in$ non va bene perchè significherebbe che serve solo 1 valore

$$\psi(x, y) = \begin{cases} 1 & se\ x \in K \land y \overset{\downarrow}{=} [0, m] \\ \uparrow & altrimenti \end{cases}$$

```
input(x,y)
if phi_x(x) termina && y in [0, m] {
  return 1
} else {
  while true
}
```

Siccome psi è parziale ricorsiva

Per smn $\exists g: \mathbb{N} \to \mathbb{N}.\ \psi(x, y) = \varphi_{g(x)}(y)$

Quindi:

$K \leq_F A$

- se $x \in K \Rightarrow \psi(x, y)\downarrow \iff y = [0, m]$

$\overset{smn}{\Rightarrow} \varphi_{g(x)}(y)\downarrow \iff \forall y \in [0, m]$

$\Rightarrow W_{g(x)} = [0, m] \Rightarrow g(x) \in A$

$\bar{K} \leq_F \bar{A}$

- se $x \notin K \Rightarrow \forall y \in \mathbb{N}.\ \psi(x, y)\uparrow$

$\overset{smn}{\Rightarrow} \forall y \in \mathbb{N}.\ \varphi_{g(x)}(y)$

$\Rightarrow W_{g(x)} = \emptyset \Rightarrow g(x) \notin A \equiv g(x) \in \bar{A}$

9. $A = \{x \mid W_x \not\subseteq 2\mathbb{N}+1\}$

Intuitivamente l'insieme è creativo.

Dimostro:

1) $A \in RE$

$$\psi_A(x) = \begin{cases} 1 & se\ x \in A \\ \uparrow & altrimenti \end{cases}$$

Bisogna fare un dovetail su tutti i numeri pari:

```
input(x)
costruisci phi_x // Esiste una procedura iterativa che la costruisce
while true {
  for j = 0 to y {
    esegui un passo di phi_x(j)
    if phi_x(j) termina return 1
    j += 2
  }
  y += 2
}
```

2) $A \in CR$:

$K \leq_F A \Leftrightarrow \bar{K} \leq_F \bar{A}$

$$\psi(x, y) = \begin{cases} 1 & se\ x \in K \wedge y \in 2\mathbb{N} \\ \uparrow & altrimenti \end{cases}$$

Siccome psi è parziale ricorsiva

Per smn $\exists g: \mathbb{N} \to \mathbb{N}.\ \psi(x, y) = \varphi_{g(x)}(y)$

- Se $x \in K \Rightarrow \exists y \in 2\mathbb{N}.\ \psi(x, y)\downarrow$

$\overset{smn}{\Rightarrow} \exists y \in 2\mathbb{N}.\ \varphi_{g(x)}(y)\downarrow$

$\Rightarrow W_{g(x)} \not\subseteq 2\mathbb{N}+1 \Rightarrow g(x) \in A$

- Se $x \notin K \Rightarrow \forall y \in 2\mathbb{N},\ \psi(x, y)\uparrow$

$\overset{smn}{\Rightarrow} \forall y \in 2\mathbb{N},\ \varphi_{g(x)}(y)\uparrow$

$\Rightarrow W_{g(x)} = \emptyset \subseteq 2\mathbb{N}+1 \Rightarrow g(x) \notin A \equiv g(x) \in \bar{A}$

10. $A = \{x \mid (\exists y.\ y \leq y^2) \Rightarrow W_x \neq \emptyset\}$

Intuitivamente l'insieme è creativo.
La condizione a sinistra è sempre vera, quindi l'insieme si può riscrivere:

$$A = \{x \mid W_x \neq \emptyset\}$$

11. $A = \{x \mid x \in 2\mathbb{N} \Rightarrow W_x \neq \emptyset\}$

1) $A \in RE$

$$\psi_A(x) = \begin{cases} 1 & \text{se } x \in A \\ \uparrow & \text{altrimenti} \end{cases}$$

```
input(x)

// Premessa dell'implicazione falsa → implicazione vera
if x % 2 ≠ 0 {
  return 1
}

...
```

$\alpha \to \beta \equiv \neg\alpha \vee \beta$

$\neg\alpha$

β

Insiemi produttivi:

1. $A = \{x \mid W_x = REC \Rightarrow \varphi_x(x^2)\downarrow\}$ $\quad REC \in \{\mathbb{N}, 2\mathbb{N}, 3\mathbb{N}, \dots\}$

Insiemi infiniti ricorsivi

$= \{x \mid \underbrace{W_x \neq REC}_{\text{non decidibile}} \vee \varphi_x(x^2)\downarrow\}$

$\bar{A} = \{x \mid W_x = REC \wedge \underbrace{\varphi_x(x^2)\uparrow}_{\text{non decidibile}}\}$

Gli insiemi sembrano entrambi produttivi

- Dimostro che A è produttivo: $\bar{K} \leq A \equiv K \leq \bar{A}$

Costruisco una funzione parziale ricorsiva psi su cui poi applicare il teorema smn

1° Tentativo

$$\psi(x,y) = \begin{cases} 1 & \text{se } x \in K \\ \uparrow & \text{altrimenti} \end{cases}$$

$x \in K \Rightarrow \varphi_x(x)\downarrow$

$\Rightarrow \forall y.\ \psi(x,y)\downarrow$

$\Rightarrow \forall y.\ \varphi_{g(x)}(y)\downarrow$

$\Rightarrow \varphi_{g(x)}(g(x)^2)\uparrow$

$\Rightarrow W_{g(x)} = \mathbb{N} \in REC$ ✗ Sbagliato perchè manca l'altra parte dell'AND

$x \notin K \Rightarrow \forall y \in \mathbb{N}.\ \psi(x,y)\uparrow$

$\Rightarrow \forall y \in \mathbb{N}.\ \varphi_{g(x)}(y)\uparrow$

$\Rightarrow \varphi_{g(x)}(g(x)^2)\uparrow$

$\Rightarrow W_{g(x)} = \emptyset \Rightarrow g(x) \in A$ L'insieme vuoto non appartiene agli INFINITI ricorsivi

2° Tentativo

$$\psi(x,y) = \begin{cases} 1 & \text{se } x \in K \wedge y \in 2\mathbb{N}+1 \\ 1 & \text{se } y = 1 \\ \uparrow & \text{altrimenti} \end{cases}$$

$x \in K \Rightarrow \varphi_x(x)\downarrow \iff y \neq 1$

$\Rightarrow \forall y. \in 2\mathbb{N}+1 / \{1\}.\ \psi(x,y)\downarrow$

$\Rightarrow \forall y. \in 2\mathbb{N}+1 / \{1\}.\ \varphi_{g(x)}(y)\downarrow$

$\Rightarrow \varphi_{g(x)}(1)\uparrow$

$\Rightarrow W_{g(x)} = \mathbb{N} / \{1\} \in REC$

$x \notin K$ Analogo al 1° Tentativo

- Dimostro che $\overline{A}$ è produttivo: $\overline{K} \leq \overline{A} \equiv K \leq A$

1° Tentativo

$$\psi(x,y) = \begin{cases} 1 & \varphi_x(x)\not\downarrow \text{ in meno di } y \text{ passi} \wedge y \in 2\mathbb{N}+1 \\ 1 & y = 1 \\ \uparrow & \text{altrimenti} \end{cases}$$

```
input(x,y)
if x == 1 {
  while true
}
...
```

quando $x \notin K \Rightarrow \varphi_x(x)\not\downarrow$ in meno di y passi è sempre vero perchè per def di k diverge sempre

$$x \notin K \Rightarrow \psi(x,y)\downarrow \iff y \in 2\mathbb{N}+1 \setminus \{1\}$$

$$\overset{smn}{\Rightarrow} \forall y \in 2\mathbb{N}+1 \setminus \{1\} .\ \varphi_{g(x)}(y)\downarrow$$

$$\Rightarrow W_{g(x)} = 2\mathbb{N}+1 \setminus \{1\} \in REC \wedge (g(x)^2 \notin W_x \iff g(x) = 1)$$

$$\Rightarrow g(x) \notin A$$

$$x \in K \Rightarrow \psi(x,y)\downarrow \iff \exists n.\ y \in [0, n-1] \cap 2\mathbb{N}+1 \setminus \{1\}$$

$$\overset{smn}{\Rightarrow} \forall y .\ \exists n,\ y \in [0, n-1] \cap 2\mathbb{N}+1 \setminus \{1\} .\ \varphi_{g(x)}(y)\downarrow$$

$$\Rightarrow W_{g(x)} = [0, n-1] \cap 2\mathbb{N}+1 \setminus \{1\} \notin REC$$

Ho dimostrato che A e il suo complemento sono produttivi

2. $D_n = \mathrm{Dom}(\psi_n) \qquad n \geq 3$

$$\psi_n(x) = \begin{cases} \overbrace{\lceil \sqrt{n+x} \rceil}^{\text{Range (inutile)}} & \overbrace{\underbrace{4x+1 \in 2\mathbb{N}+1}_{\text{sempre vero}} \wedge 4x+1 \in W_x \text{ non deciso in meno di } n \text{ passi}}^{\text{dominio}} \\ \uparrow & \text{altrimenti} \end{cases}$$

$\equiv$ $\varphi_x(4x+1)\uparrow$ in meno di n passi

Il dominio è dettato solo dalla seconda condizione

$$X_n = \{ \lceil \sqrt{x} \rceil \mid n \in W_x \}$$

Studiare il seguente insieme e il suo complemento:

$$S = \bigcap_{n \in \mathbb{R}} X_n \qquad R = \bigcap_{n \geq 3} D_n$$

$D_3 = \{ x \mid 4x+1 \notin W_x \text{ in meno di 3 passi} \}$ (Programmi che terminano in più di 3 passi)

$D_4 = \{ x \mid 4x+1 \notin W_x \text{ in meno di 4 passi} \}$ (Programmi che terminano in più di 4 passi)

⋮

Quindi:

$$R = \bigcap_{n \geq 3} D_n = \{x \mid 4x+1 \notin W_x\} \qquad R \in PR \ \wedge \ \overline{R} \in CR$$

$$S = \bigcap_{n \in R} X_n = \{\lceil \sqrt{x} \rceil \mid R \subseteq W_x$$