Esercizi Risolti

Pumping Lemma e Linguaggi non Regolari: casi studio avanzati

Integrazione al Tutorato 4

Gabriel Rovesti

Corso di Laurea in Informatica - Università degli Studi di Padova

Anno Accademico 2024-2025

Contents

1 Introduzione **2**

2 Linguaggi con Bilanciamento di Parentesi **3**
2.1 Linguaggio di Parentesi Bilanciate . . . 3
2.2 Linguaggio Dyck con più Tipi di Parentesi . . . 4

3 Linguaggi con Proprietà Combinatorie **5**
3.1 Linguaggio di Stringhe con Prefisso e Suffisso Uguali . . . 5
3.2 Linguaggio di Stringhe XX . . . 6

4 Linguaggi con Proprietà Aritmetiche **7**
4.1 Linguaggio di Potenze . . . 7
4.2 Linguaggio con Divisibilità . . . 8

5 Linguaggi con Pattern Matching Complessi **10**
5.1 Linguaggio di Pattern non Sovrapposti . . . 10
5.2 Linguaggio con Conteggio di Pattern . . . 11

6 Tecniche Avanzate e Casi Particolari **12**
6.1 Dimostrazione per Contraddizione delle Proprietà di Chiusura . . . 12
6.2 Utilizzo del Lemma di Ogden . . . 13

7 Applicazioni Pratiche **14**
7.1 Riconoscimento di Pattern Complessi . . . 14
7.2 Analisi di Protocolli di Comunicazione . . . 15

8 Conclusioni e Suggerimenti Metodologici **16**

1 Introduzione

Questo documento integra il materiale presentato nel Tutorato 4 con ulteriori esempi di applicazione del Pumping Lemma per dimostrare la non regolarità di linguaggi particolarmente interessanti o complessi. Gli esercizi sono presentati con soluzioni dettagliate, analisi dei passaggi critici e suggerimenti metodologici.

Concetto chiave

Ricordiamo l'enunciato del Pumping Lemma: Se L è un linguaggio regolare, allora esiste una costante $p > 0$ tale che ogni stringa $s \in L$ con $|s| \geq p$ può essere scomposta in $s = xyz$ dove:

1. $|y| > 0$ (la parte "pompabile" non è vuota)
2. $|xy| \leq p$ (i primi due pezzi insieme hanno lunghezza al massimo p)
3. $\forall i \geq 0$, $xy^iz \in L$ (la stringa ottenuta ripetendo y un numero arbitrario di volte appartiene ancora a L)

2 Linguaggi con Bilanciamento di Parentesi

2.1 Linguaggio di Parentesi Bilanciate

Procedimento di risoluzione

Esercizio 1. Dimostrare che il linguaggio di parentesi bilanciate $L = \{w \in \{(,)\}^* \mid \text{le parentesi sono bilanciate}\}$ non è regolare.
Soluzione:

1. Assumiamo per assurdo che L sia regolare con pumping length $p > 0$
2. Consideriamo la stringa $w = (^p)^p \in L$ (p parentesi aperte seguite da p parentesi chiuse)
3. Per qualsiasi decomposizione $w = xyz$ con $|y| > 0$ e $|xy| \leq p$:
   - Poiché $|xy| \leq p$, la sottostringa y deve essere contenuta interamente nella prima parte della stringa, quella delle parentesi aperte
   - Quindi $y = (^k$ per qualche $k > 0$
4. Consideriamo $i = 2$: $xy^2z = (^{p+k})^p$
5. Questa stringa ha $p + k$ parentesi aperte e solo p parentesi chiuse
6. Quindi le parentesi non sono bilanciate e $xy^2z \notin L$
7. Questo contraddice l'ipotesi che L sia regolare

Nota: Questo linguaggio è un esempio classico di linguaggio context-free non regolare. La sua grammatica è semplice: $S \to \epsilon \mid (S) \mid SS$, ma richiede una pila per tenere traccia delle parentesi aperte, una capacità che gli automi a stati finiti non possiedono.

Suggerimento

È possibile generalizzare questa dimostrazione a linguaggi di parentesi multiple $\{(,),[,],\{,\}\}$ o a linguaggi come $\{a^nb^nc^n \mid n \geq 0\}$, che richiedono di contare e confrontare il numero di occorrenze di più di un simbolo.

2.2 Linguaggio Dyck con più Tipi di Parentesi

Procedimento di risoluzione

Esercizio 2. Dimostrare che il linguaggio Dyck con due tipi di parentesi $L = \{w \in \{(,),[,]\}^* \mid \text{le parentesi sono correttamente annidate}\}$ non è regolare.
Soluzione:

1. Assumiamo per assurdo che L sia regolare con pumping length $p > 0$
2. Consideriamo la stringa $w = (^p[^p]^p)^p \in L$
3. Per qualsiasi decomposizione $w = xyz$ con $|y| > 0$ e $|xy| \leq p$:
   - Poiché $|xy| \leq p$, la sottostringa y deve essere contenuta interamente nella prima parte della stringa, quella delle parentesi tonde aperte
   - Quindi $y = (^k$ per qualche $k > 0$
4. Consideriamo $i = 0$: $xy^0z = (^{p-k}[^p]^p)^p$
5. Questa stringa ha $p - k$ parentesi tonde aperte iniziali, ma sempre p parentesi tonde chiuse finali
6. Il bilanciamento è stato alterato e $xy^0z \notin L$
7. Questo contraddice l'ipotesi che L sia regolare

Osservazione: Abbiamo scelto una stringa che "annida" un tipo di parentesi dentro l'altro per forzare una struttura che un automa a stati finiti non può riconoscere. Qualsiasi variazione nel numero di parentesi compromette il corretto annidamento.

3 Linguaggi con Proprietà Combinatorie

3.1 Linguaggio di Stringhe con Prefisso e Suffisso Uguali

Procedimento di risoluzione

Esercizio 3.
Sia $L = \{w \in \{a,b\}^* \mid w \text{ ha un prefisso uguale al suffisso di lunghezza } \lfloor |w|/3 \rfloor\}$. Dimostrare che L non è regolare.
Soluzione:

1. Assumiamo per assurdo che L sia regolare con pumping length $p > 0$
2. Consideriamo la stringa $w = a^p b^{3p} a^p \in L$ (poiché $|w| = 5p$, $\lfloor |w|/3 \rfloor = \lfloor 5p/3 \rfloor \geq p$ per $p \geq 3$)
3. Per qualsiasi decomposizione $w = xyz$ con $|y| > 0$ e $|xy| \leq p$:
   - Poiché $|xy| \leq p$, la sottostringa y deve essere contenuta interamente nel prefisso a^p
   - Quindi $y = a^k$ per qualche $k > 0$
4. Consideriamo $i = 2$: $xy^2z = a^{p+k} b^{3p} a^p$
5. Per questa stringa, $|xy^2z| = 5p+k$ e $\lfloor |xy^2z|/3 \rfloor = \lfloor (5p+k)/3 \rfloor \geq p + \lfloor k/3 \rfloor > p$ per $k > 0$
6. Il prefisso e il suffisso di lunghezza $\lfloor |xy^2z|/3 \rfloor > p$ non possono essere uguali, poiché il prefisso contiene solo a mentre il suffisso contiene alcuni b
7. Quindi $xy^2z \notin L$, contraddicendo l'ipotesi che L sia regolare

Nota: Questo esempio mostra come le proprietà combinatorie che richiedono di confrontare parti distanti di una stringa siano tipicamente oltre le capacità degli automi a stati finiti.

3.2 Linguaggio di Stringhe XX

Procedimento di risoluzione

Esercizio 4. Sia $L = \{ww \mid w \in \{a,b\}^+\}$, il linguaggio delle stringhe che sono l'esatta duplicazione di una stringa non vuota. Dimostrare che L non è regolare.
Soluzione:

1. Assumiamo per assurdo che L sia regolare con pumping length $p > 0$
2. Consideriamo la stringa $w = (ab)^p(ab)^p \in L$, che è la duplicazione di $(ab)^p$
3. Per qualsiasi decomposizione $w = xyz$ con $|y| > 0$ e $|xy| \leq p$:
   - Poiché $|xy| \leq p < 2p = |(ab)^p|$, la sottostringa y deve essere contenuta interamente nella prima metà di w
   - Date le dimensioni, y deve essere della forma $(ab)^j$ o $b(ab)^{j-1}a$ o simili per qualche $j > 0$
4. Consideriamo $i = 2$ e supponiamo $y = (ab)^j$:
   - $xy^2z = (ab)^{p+j}(ab)^p$
   - Questa stringa non può essere scritta come uu per alcun u, poiché la prima metà $(ab)^{p+j}$ è più lunga della seconda metà $(ab)^p$
5. Quindi $xy^2z \notin L$, contraddicendo l'ipotesi che L sia regolare

Variante: Se avessimo scelto $y = b(ab)^{j-1}a$, la dimostrazione sarebbe stata più complessa, ma il risultato sarebbe lo stesso: la stringa pompata non mantiene la struttura di duplicazione esatta.

Suggerimento

Per linguaggi che richiedono confronti tra parti distanti della stringa, è sempre utile scegliere una stringa di test che forzi il "pompaggio" in una parte critica che altererà la proprietà fondamentale del linguaggio.

4 Linguaggi con Proprietà Aritmetiche

4.1 Linguaggio di Potenze

Procedimento di risoluzione

Esercizio 5. Sia $L = \{a^{m^n} \mid m, n \geq 2\}$, il linguaggio delle stringhe la cui lunghezza è una potenza di un numero. Dimostrare che L non è regolare.
Soluzione:

1. Assumiamo per assurdo che L sia regolare con pumping length $p > 0$
2. Scegliamo $m = 2$ e n tale che $2^n > p$ (ad esempio, $n = \lceil \log_2 p \rceil + 1$)
3. Consideriamo la stringa $w = a^{2^n} \in L$
4. Per qualsiasi decomposizione $w = xyz$ con $|y| > 0$ e $|xy| \leq p$:
   - Poiché la stringa contiene solo a, abbiamo $y = a^k$ per qualche $k > 0$
5. Consideriamo $i = 2$: $xy^2z = a^{2^n+k}$
6. Dobbiamo dimostrare che $2^n + k$ non è della forma $m'^{n'}$ per alcun $m', n' \geq 2$:
   - Se $2^n + k = m'^{n'}$, allora $m'^{n'} - 2^n = k$
   - Poiché $k \leq p < 2^n$, abbiamo $0 < k < 2^n$
   - Per qualsiasi $m' \geq 2$ e $n' \geq 2$, se $m'^{n'} > 2^n$ allora $m'^{n'} - 2^n \geq 2^n$ (perché la differenza tra potenze successive cresce rapidamente)
   - Se $m'^{n'} < 2^n$, allora $m'^{n'} - 2^n < 0$
   - Quindi non esiste alcuna coppia m', n' tale che $m'^{n'} - 2^n = k$ con $0 < k < 2^n$
7. Quindi $xy^2z \notin L$, contraddicendo l'ipotesi che L sia regolare

Nota: Questo esempio richiede una comprensione delle proprietà delle potenze e della loro crescita. La dimostrazione sfrutta il fatto che non ci sono "potenze adiacenti" tra loro per numeri sufficientemente grandi.

4.2 Linguaggio con Divisibilità

Procedimento di risoluzione

Esercizio 6. Sia $L = \{a^n b^m \mid n, m \geq 1, n \text{ è divisibile per } m\}$. Dimostrare che L non è regolare.

Soluzione:

1. Assumiamo per assurdo che L sia regolare con pumping length $p > 0$
2. Consideriamo la stringa $w = a^{p!}b^p \in L$ (dove $p!$ indica il fattoriale di p)
3. Poiché $p!$ è divisibile per ogni intero da 1 a p, incluso p, abbiamo $w \in L$
4. Per qualsiasi decomposizione $w = xyz$ con $|y| > 0$ e $|xy| \leq p$:
   - Poiché $|xy| \leq p$, la sottostringa y deve essere contenuta interamente nella parte $a^{p!}$
   - Quindi $y = a^k$ per qualche $k > 0$
5. Consideriamo $i = 0$: $xy^0z = a^{p!-k}b^p$
6. Dobbiamo dimostrare che $p! - k$ non è divisibile per p per qualche valore di k con $1 \leq k \leq p$:
   - Sappiamo che $p!$ è divisibile per p, quindi $p! = pq$ per qualche q
   - Se k non è divisibile per p, allora $p! - k = pq - k$ non è divisibile per p
   - Poiché $k \leq p$, se $k \neq p$ allora k non è divisibile per p
   - Se $y = a^p$, possiamo considerare $i = 2$ ottenendo $xy^2z = a^{p!+p}b^p$
   - Poiché $p! + p = p(q + 1)$, il numero di a rimane divisibile per p
   - Se $1 \leq k < p$, allora k non è divisibile per p, e possiamo usare $i = 0$
7. In generale, per ogni possibile decomposizione, possiamo trovare un valore di i tale che $xy^iz \notin L$
8. Questo contraddice l'ipotesi che L sia regolare

Osservazione: Questo esempio mostra come le proprietà aritmetiche come la divisibilità non possano essere verificate da un automa a stati finiti quando i numeri coinvolti non hanno un limite superiore.

Errore comune

Nella dimostrazione precedente, è importante notare che non basta mostrare che esiste una decomposizione per cui il pompaggio porta fuori dal linguaggio. Bisogna dimostrare che per *qualsiasi* decomposizione che rispetti le condizioni del Pumping Lemma, esiste un valore di i tale che $xy^iz \notin L$.

5 Linguaggi con Pattern Matching Complessi

5.1 Linguaggio di Pattern non Sovrapposti

Procedimento di risoluzione

Esercizio 7.
Sia $L = \{w \in \{a, b\}^* \mid w \text{ contiene il pattern } aba \text{ esattamente una volta}\}$. Dimostrare che L non è regolare.
Soluzione:

1. Assumiamo per assurdo che L sia regolare con pumping length $p > 0$
2. Consideriamo la stringa $w = b^p abab^p \in L$
3. Per qualsiasi decomposizione $w = xyz$ con $|y| > 0$ e $|xy| \leq p$:
   - Poiché $|xy| \leq p$, la sottostringa y deve essere contenuta interamente nel prefisso b^p
   - Quindi $y = b^k$ per qualche $k > 0$
4. Consideriamo $i = 2$: $xy^2z = b^{p+k}abab^p$
5. Questa stringa contiene ancora esattamente una occorrenza di aba, quindi $xy^2z \in L$
6. La scelta non ha portato a una contraddizione, dobbiamo riconsiderare la nostra strategia

Proviamo con una stringa diversa:

1. Consideriamo $w' = b^p(aba)^2b^p$, che non appartiene a L perché contiene due occorrenze di aba
2. Modifichiamo la stringa aggiungendo un separatore: $w = b^p abac^p abab^p$
3. Questa stringa può essere vista come appartenente al linguaggio
   $L' = \{w \in \{a, b, c\}^* \mid w \text{ contiene il pattern } aba$
   $\text{esattamente due volte e le occorrenze sono separate da almeno una } c\}$
4. Se L' non è regolare, allora L non può essere regolare (si può dimostrare con una riduzione)
5. Per qualsiasi decomposizione $w = xyz$ con $|y| > 0$ e $|xy| \leq p$:
   - La sottostringa y deve essere contenuta interamente nel prefisso b^p
   - Quindi $y = b^k$ per qualche $k > 0$
6. Consideriamo $i = 0$: $xy^0z = b^{p-k}abac^p abab^p$
7. Consideriamo ora un automa a stati finiti che riconosce il linguaggio di stringhe con esattamente una occorrenza di aba
8. Un tale automa deve cambiare stato in modo permanente dopo aver visto la prima occorrenza di aba
9. Quando legge $b^{p-k}aba$ ha già cambiato stato, e non può tornare allo stato precedente, quindi non può riconoscere la seconda occorrenza di aba come valida ed L non può essere regolare

Concetto chiave

Quando il Pumping Lemma diretto non porta a una contraddizione evidente, è possibile utilizzare altre proprietà dei linguaggi regolari, come la chiusura rispetto a operazioni come l'intersezione con linguaggi regolari o la sostituzione di simboli.

5.2 Linguaggio con Conteggio di Pattern

Procedimento di risoluzione

Esercizio 8. Sia $L = \{w \in \{a,b\}^* \mid w \text{ contiene un ugual numero di occorrenze delle sottostringhe } aa \text{ e } bb\}$. Dimostrare che L non è regolare.

Soluzione:

1. Assumiamo per assurdo che L sia regolare con pumping length $p > 0$
2. Consideriamo la stringa $w = (aa)^p(bb)^p \in L$ (che contiene esattamente p occorrenze sia di aa che di bb)
3. Per qualsiasi decomposizione $w = xyz$ con $|y| > 0$ e $|xy| \leq p$:
   - Poiché $|xy| \leq p < 2p = |(aa)^p|$, la sottostringa y deve essere contenuta interamente nella prima parte $(aa)^p$
   - Analizzando le possibilità, y può essere $(aa)^j$ per qualche $j > 0$, o può attraversare i confini delle sottostringhe aa
4. Consideriamo il caso più semplice: $y = (aa)^j$ per qualche $j > 0$
5. Scegliamo $i = 2$: $xy^2z = (aa)^{p+j}(bb)^p$
6. Questa stringa contiene $p + j$ occorrenze di aa e solo p occorrenze di bb
7. Quindi $xy^2z \notin L$, contraddicendo l'ipotesi che L sia regolare

Nota: Se y attraversa i confini, ad esempio $y = a(aa)^{j-1}a$, dobbiamo contare attentamente le occorrenze di aa nella stringa pompata. In questo caso, la stringa pompata conterrebbe comunque un numero diverso di aa rispetto a bb.

Suggerimento

Per linguaggi che coinvolgono il conteggio di pattern, una buona strategia è costruire una stringa con un equilibrio esatto dei pattern, e poi dimostrare che qualsiasi pompaggio disturberà questo equilibrio.

6 Tecniche Avanzate e Casi Particolari

6.1 Dimostrazione per Contraddizione delle Proprietà di Chiusura

Procedimento di risoluzione

Esercizio 9. Dimostrare che il linguaggio $L = \{a^i b^j c^k \mid i, j, k \geq 0 \text{ e se } i = j \text{ allora } j = k\}$ non è regolare utilizzando le proprietà di chiusura dei linguaggi regolari.
Soluzione:

1. Consideriamo il linguaggio $L_1 = \{a^n b^n c^n \mid n \geq 0\}$, che sappiamo non essere regolare
2. Consideriamo anche il linguaggio $L_2 = \{a^i b^j c^k \mid i = j \text{ o } j \neq k\}$
3. Osserviamo che $L_2 = L$ (è la stessa definizione riscritta in modo logicamente equivalente)
4. Consideriamo il complemento di L_2: $\overline{L_2} = \{a^i b^j c^k \mid i \neq j \text{ e } j = k\}$
5. Se L_2 fosse regolare, anche $\overline{L_2}$ sarebbe regolare (i linguaggi regolari sono chiusi rispetto al complemento)
6. Consideriamo $L_3 = \{a^i b^j c^k \mid i = j = k\} = L_1$
7. Consideriamo anche $L_4 = \{a^i b^j c^k \mid i \neq j\}$
8. Allora $L_3 \cap L_4 = \emptyset$ (insiemi disgiunti)
9. E $\overline{L_2} = \{a^i b^j c^k \mid i \neq j \text{ e } j = k\} = L_4 \cap \{a^i b^j c^k \mid j = k\}$
10. Se $\overline{L_2}$ fosse regolare, allora $\overline{L_2} \cup L_3 = \{a^i b^j c^k \mid (i \neq j \text{ e } j = k) \text{ o } (i = j = k)\} = \{a^i b^j c^k \mid j = k\}$ sarebbe regolare (i linguaggi regolari sono chiusi rispetto all'unione)
11. Ma $\{a^i b^j c^k \mid j = k\} \cap \{a^i b^j c^k \mid i = j\} = \{a^i b^j c^k \mid i = j = k\} = L_1$
12. Se entrambi fossero regolari, anche la loro intersezione L_1 sarebbe regolare
13. Ma sappiamo che L_1 non è regolare, quindi abbiamo una contraddizione
14. Di conseguenza, L non può essere regolare

Osservazione: Questa dimostrazione è più complessa e utilizza le proprietà di chiusura dei linguaggi regolari rispetto a operazioni come complemento, unione e intersezione. È un approccio alternativo quando l'applicazione diretta del Pumping Lemma risulta difficile.

Concetto chiave

I linguaggi regolari sono chiusi rispetto a operazioni come unione, intersezione, complemento, concatenazione e chiusura di Kleene. Questa proprietà può essere sfruttata per dimostrare la non regolarità di un linguaggio riducendolo a un caso noto.

6.2 Utilizzo del Lemma di Ogden

Procedimento di risoluzione

Esercizio 10. Il Lemma di Ogden è una generalizzazione del Pumping Lemma che permette di "marcare" alcune posizioni nella stringa e richiede che la parte pompabile contenga almeno una posizione marcata. Utilizzando questo lemma, dimostrare che il linguaggio $L = \{a^n b^n c^m \mid n, m \geq 1\}$ non è regolare.
Soluzione:

1. Il Lemma di Ogden afferma che se L è regolare, allora esiste una costante $p > 0$ tale che per ogni stringa $z \in L$ e per ogni modo di marcare almeno p posizioni in z, z può essere decomposta come $z = uvwxy$ con:
   - $|vwx| \leq p$
   - v e x non possono essere entrambi vuoti
   - vwx contiene al massimo p posizioni marcate
   - Per ogni $i \geq 0$, $uv^iwx^iy \in L$
2. Consideriamo la stringa $z = a^p b^p c^p \in L$
3. Marchiamo tutte le posizioni delle a e delle b (cioè, le prime $2p$ posizioni)
4. Per qualsiasi decomposizione $z = uvwxy$ secondo il Lemma di Ogden:
   - Poiché ci sono $2p$ posizioni marcate e vwx può contenere al massimo p posizioni marcate, o u o y (o entrambi) devono contenere posizioni marcate
   - Se y contiene posizioni marcate, queste possono essere solo a o b
   - Se y contiene a, allora con $i = 2$ otteniamo una stringa con più a che b
   - Se y contiene b, allora con $i = 2$ otteniamo una stringa con più b che a
   - In entrambi i casi, la stringa pompata non appartiene a L
5. Questo contraddice l'ipotesi che L sia regolare

Nota: Il Lemma di Ogden è particolarmente utile per dimostrare la non regolarità di linguaggi per cui il Pumping Lemma standard è difficile da applicare. Permette un controllo più preciso su quali parti della stringa vengono pompate.

7 Applicazioni Pratiche

7.1 Riconoscimento di Pattern Complessi

Procedimento di risoluzione

Esercizio 11. Consideriamo un sistema che deve verificare se un input è un indirizzo email valido. Un indirizzo email semplificato può essere rappresentato dal linguaggio $L = \{w_1@w_2 \mid w_1, w_2 \in \{a, \ldots, z, 0, \ldots, 9\}^+, w_2 \text{ contiene almeno un punto } .\}$. Dimostrare che questo linguaggio non è regolare.

Soluzione: Per questo esercizio, dobbiamo modificare leggermente la definizione per includere il punto nell'alfabeto: $L = \{w_1@w_2 \mid w_1 \in \{a, \ldots, z, 0, \ldots, 9\}^+, w_2 \in \{a, \ldots, z, 0, \ldots, 9, .\}^+, w_2 \text{ contiene almeno un } .\}$

1. Assumiamo per assurdo che L sia regolare con pumping length $p > 0$
2. Consideriamo la stringa $w = a^p@a^p.a^p \in L$
3. Per qualsiasi decomposizione $w = xyz$ con $|y| > 0$ e $|xy| \leq p$:
   - Poiché $|xy| \leq p$, la sottostringa y deve essere contenuta interamente nel prefisso a^p
   - Quindi $y = a^k$ per qualche $k > 0$
4. Consideriamo $i = 0$: $xy^0z = a^{p-k}@a^p.a^p$
5. Questa stringa è ancora della forma $w_1@w_2$ dove w_2 contiene un punto
6. Quindi $xy^0z \in L$, e non otteniamo una contraddizione
7. Dobbiamo rivedere la nostra strategia

Proviamo con un altro approccio, utilizzando l'intersezione con un linguaggio regolare:

1. Consideriamo il linguaggio regolare $R = \{a^n@a^n.a^m \mid n, m \geq 1\}$
2. L'intersezione $L \cap R = \{a^n@a^n.a^m \mid n, m \geq 1\}$ dovrebbe essere regolare se L è regolare
3. Ma $L \cap R$ è essenzialmente equivalente a $\{a^nb^n \mid n \geq 1\}$ (con una codifica adeguata)
4. Poiché $\{a^nb^n \mid n \geq 1\}$ non è regolare, neanche $L \cap R$ può essere regolare
5. Questo contraddice l'ipotesi che L sia regolare

Implicazioni pratiche: Questo risultato suggerisce che un validatore di email completamente accurato non può essere implementato utilizzando solo espressioni regolari, ma richiede un parser più potente (come un parser context-free).

Suggerimento

Nella pratica, molte applicazioni utilizzano espressioni regolari per validare gli indirizzi email, ma queste implementazioni sono solo approssimazioni e non possono catturare tutte le complessità di uno standard completo come RFC 5322.

7.2 Analisi di Protocolli di Comunicazione

Procedimento di risoluzione

Esercizio 12. Un semplice protocollo di comunicazione richiede che ogni messaggio inviato riceva una conferma. Il linguaggio delle tracce valide può essere modellato come $L = \{(sm)^n \mid s \text{ rappresenta un invio}, m \text{ rappresenta una conferma}, n \geq 0\}$. Questo linguaggio è regolare. Consideriamo ora un protocollo più complesso che permette invii multipli prima delle conferme, ma richiede che ogni invio riceva eventualmente una conferma. Il linguaggio diventa $L' = \{w \in \{s, m\}^* \mid \text{ ogni } s \text{ in } w \text{ è seguito da una } m, \text{ e il numero totale di } s \text{ e } m \text{ è uguale}\}$. Dimostrare che L' non è regolare.

Soluzione:

1. Assumiamo per assurdo che L' sia regolare con pumping length $p > 0$
2. Consideriamo la stringa $w = s^p m^p \in L'$
3. Per qualsiasi decomposizione $w = xyz$ con $|y| > 0$ e $|xy| \leq p$:
   - Poiché $|xy| \leq p$, la sottostringa y deve essere contenuta interamente nella prima parte s^p
   - Quindi $y = s^k$ per qualche $k > 0$
4. Consideriamo $i = 2$: $xy^2z = s^{p+k}m^p$
5. Questa stringa contiene $p + k$ occorrenze di s e solo p occorrenze di m
6. Quindi non ogni s può essere seguito da una m corrispondente, o il numero totale di s e m non è uguale
7. In entrambi i casi, $xy^2z \notin L'$
8. Questo contraddice l'ipotesi che L' sia regolare

Implicazioni pratiche: Questa dimostrazione suggerisce che l'analisi di protocolli che richiedono di tenere traccia di eventi correlati (come invii e conferme) non può essere completamente realizzata con automi a stati finiti, ma richiede modelli computazionali più potenti.

8 Conclusioni e Suggerimenti Metodologici

Concetto chiave

Riassumiamo le principali strategie per dimostrare la non regolarità di un linguaggio:

1. **Applicazione diretta del Pumping Lemma**:
   - Scegliere una stringa appropriata $w \in L$ con $|w| \geq p$
   - Analizzare tutte le possibili decomposizioni $w = xyz$
   - Trovare un valore di i tale che $xy^iz \notin L$
2. **Utilizzo delle proprietà di chiusura**:
   - Dimostrare che se L fosse regolare, allora anche un linguaggio noto come non regolare lo sarebbe
   - Utilizzare operazioni come unione, intersezione, complemento, ecc.
3. **Lemma di Ogden e altre generalizzazioni**:
   - Utilizzare versioni più potenti del Pumping Lemma per casi complessi
   - Marcare posizioni specifiche nella stringa per forzare decomposizioni particolari

Suggerimento

Quando si affronta un nuovo linguaggio, porsi le seguenti domande:

- Quali sono le proprietà fondamentali che caratterizzano il linguaggio?
- Quale struttura deve avere una stringa per appartenere al linguaggio?
- Quale proprietà verrebbe alterata pompando una parte della stringa?
- Esiste una stringa "critica" che forza l'uso di memoria illimitata?
- È possibile ridurre il problema a un caso noto di linguaggio non regolare?

Errore comune

Evitare questi errori comuni:

- Dimostrare che *esiste* una decomposizione problematica, invece di dimostrare che *tutte* le decomposizioni possibili portano a una contraddizione
- Confondere la direzione logica del Pumping Lemma (è una condizione necessaria ma non sufficiente)
- Scegliere stringhe troppo semplici che non forzano decomposizioni significative
- Non considerare tutti i possibili casi di decomposizione
- Concludere erroneamente che un linguaggio è regolare solo perché non si è riusciti a dimostrare che non lo è

La teoria dei linguaggi formali, e in particolare lo studio dei linguaggi non regolari, ha importanti applicazioni pratiche in informatica, dalla progettazione di compilatori all'analisi di protocolli, dalla validazione di input all'elaborazione di testi. Padroneggiare queste tecniche non solo migliora la comprensione teorica, ma fornisce strumenti potenti per affrontare problemi pratici di progettazione e analisi di sistemi.