Teorema di Fagin

Stefano Pessotto

14 gennaio 2022

Università degli studi di Udine - Logica per L'informatica

Introduzione

Significato del teorema di Fagin

Teorema di Fagin

$\exists$SO cattura NP

Traduzione

- Per ogni formula $\Phi \in \exists$SO, verificare se una struttura soddisfa Φ è un problema NP;
- Ogni proprietà P che può essere valutata su strutture finite con complessità NP, è esprimibile in $\exists$SO.

Significato del teorema di Fagin

Teorema di Fagin

$$\exists\text{SO cattura NP}$$

Traduzione

- Per ogni formula $\Phi \in \exists\text{SO}$, verificare se una struttura soddisfa Φ è un problema NP;
- Ogni proprietà P che può essere valutata su strutture finite con complessità NP, è esprimibile in $\exists\text{SO}$.

Significato del teorema di Fagin

Teorema di Fagin

$$\exists\text{SO cattura NP}$$

Traduzione

- Per ogni formula $\Phi \in \exists\text{SO}$, verificare se una struttura soddisfa Φ è un problema NP;
- Ogni proprietà P che può essere valutata su strutture finite con complessità NP, è esprimibile in $\exists\text{SO}$.

Dimostrazione

Dimostrazione (1)

Dimostriamo che verificare se $\mathcal{I} \models \phi \in \exists\text{SO}$ è in NP.

Sia $\Phi = \exists S_1 \exists S_2 ... \exists S_n \varphi$, dove $\varphi \in FO$.
Costruiamo una MdT non deterministica N che:

- prende in input una struttura $\mathcal{I}$
- sceglie non deterministicamente $S_1, S_2, .., S_n$
- verifica se $\mathcal{I} \models \varphi(S_1, S_2, .., S_n)$.

Poiché la verifica $\mathcal{I} \models \varphi \in FO$, con φ fissata e $rg(\varphi) = k$, richiede tempo $\mathcal{O}((|\mathcal{I}| + |S_1| + .. + |S_n|)^k)$, N opera in tempo polinomiale.
Quindi determinare se una struttura soddisfi una formula $\Phi \in \exists SO$ è un problema NP. □

Dimostrazione (1)

Dimostriamo che verificare se $\mathcal{I} \models \phi \in \exists\text{SO}$ è in NP.

Sia $\Phi = \exists S_1 \exists S_2 ... \exists S_n \varphi$, dove $\varphi \in FO$.
Costruiamo una MdT non deterministica N che:

- prende in input una struttura $\mathcal{I}$
- sceglie non deterministicamente $S_1, S_2, .., S_n$
- verifica se $\mathcal{I} \models \varphi(S_1, S_2, .., S_n)$.

Poiché la verifica $\mathcal{I} \models \varphi \in FO$, con φ fissata e $rg(\varphi) = k$, richiede tempo $\mathcal{O}((|\mathcal{I}| + |S_1| + .. + |S_n|)^k)$, N opera in tempo polinomiale.
Quindi determinare se una struttura soddisfi una formula $\Phi \in \exists SO$ è un problema NP. □

Dimostrazione (1)

Dimostriamo che verificare se $\mathcal{I} \models \phi \in \exists\text{SO}$ è in NP.

Sia $\Phi = \exists S_1 \exists S_2 ... \exists S_n \varphi$, dove $\varphi \in FO$.
Costruiamo una MdT non deterministica N che:

- prende in input una struttura $\mathcal{I}$
- sceglie non deterministicamente $S_1, S_2, .., S_n$
- verifica se $\mathcal{I} \models \varphi(S_1, S_2, .., S_n)$.

Poiché la verifica $\mathcal{I} \models \varphi \in FO$, con φ fissata e $rg(\varphi) = k$, richiede tempo $\mathcal{O}((|\mathcal{I}| + |S_1| + .. + |S_n|)^k)$, N opera in tempo polinomiale.
Quindi determinare se una struttura soddisfi una formula $\Phi \in \exists SO$ è un problema NP. $\square$

Dimostrazione (2)

Dimostriamo che dato un problema P codificato in un linguaggio L e decidibile su strutture finite con complessità NP, P è esprimibile in $\exists SO$.

Sia $N = (Q, \Sigma, \Delta, \delta, q_o, Q_a, Q_r)$ la 1-MdT non deterministica che, data la codifica di una struttura $\mathcal{I}$ nel linguaggio L, determina se P vale in $\mathcal{I}$ in tempo polinomiale $|\mathcal{I}|^k$.

Supponiamo, senza perdita di generalità, $\Sigma = \{0, 1\}$.

Codifichiamo la struttura $\mathcal{I}$ specificando il numero di elementi nel dominio e introducendo delle stringhe di dimensione n^{m_i} per ogni predicato R_i di arietà m_i, dove n è la dimensione del dominio.

Nel caso di grafi, la codifica è data dalla stringa $0^n 1s$, dove $|s| = n^2$ e

$$0 \leq u, v \leq n-1, (u, v) \in E \iff \text{s contiene 1 in posizione } u \cdot n + v$$

Dimostrazione (2)

Dimostriamo che dato un problema P codificato in un linguaggio L e decidibile su strutture finite con complessità NP, P è esprimibile in $\exists SO$.

Sia $N = (Q, \Sigma, \Delta, \delta, q_o, Q_a, Q_r)$ la 1-MdT non deterministica che, data la codifica di una struttura $\mathcal{I}$ nel linguaggio L, determina se P vale in $\mathcal{I}$ in tempo polinomiale $|\mathcal{I}|^k$.

Supponiamo, senza perdita di generalità, $\Sigma = \{0, 1\}$.

Codifichiamo la struttura $\mathcal{I}$ specificando il numero di elementi nel dominio e introducendo delle stringhe di dimensione n^{m_i} per ogni predicato R_i di arietà m_i, dove n è la dimensione del dominio.

Nel caso di grafi, la codifica è data dalla stringa $0^n 1s$, dove $|s| = n^2$ e

$0 \leq u, v \leq n-1, (u, v) \in E \iff$ s contiene 1 in posizione $u \cdot n + v$

Dimostrazione (2)

Sappiamo che:

- la dimensione della codifica di $\mathcal{I}$ è polinomialmente correlata al dominio dell'interpretazione;
- N termina in tempo n^k, modificando quindi al più n^k celle del nastro.

Possiamo quindi esprimere un istante di tempo e una posizione del nastro tramite una k-pupla nel seguente modo:

- $\vec{t} = (t_1, .., t_k)$ codifica l'elemento $\sum_{i=1}^{k} t_i \cdot n^{k-i}$ nell'ordine lessicografico fra le k-puple corrispondenti ai passi di computazione;
- $\vec{p} = (p_1, .., p_k)$ codifica l'elemento $\sum_{i=1}^{k} p_i \cdot n^{k-i}$ nell'ordine lessicografico fra le k-puple corrispondenti alle posizioni del nastro.

Dimostrazione (2)

Sappiamo che:

- la dimensione della codifica di $\mathcal{I}$ è polinomialmente correlata al dominio dell'interpretazione;
- N termina in tempo n^k, modificando quindi al più n^k celle del nastro.

Possiamo quindi esprimere un istante di tempo e una posizione del nastro tramite una k-pupla nel seguente modo:

- $\vec{t} = (t_1, .., t_k)$ codifica l'elemento $\sum_{i=1}^{k} t_i \cdot n^{k-i}$ nell'ordine lessicografico fra le k-puple corrispondenti ai passi di computazione;
- $\vec{p} = (p_1, .., p_k)$ codifica l'elemento $\sum_{i=1}^{k} p_i \cdot n^{k-i}$ nell'ordine lessicografico fra le k-puple corrispondenti alle posizioni del nastro.

Dimostrazione (2)

Costruiamo la formula $\Phi \in \exists SO$:

$$\Phi = \exists < \exists T_0 \exists T_1 \exists T_\sqcup \exists H_{q_0} .. \exists H_{q_m} \Psi$$

dove

- $<$ è un ordine lineare;
- $\forall c \in \Delta$, il predicato $T_c(\vec{p}, \vec{t})$ indica che la posizione $\vec{p}$ del nastro, al tempo $\vec{t}$, contiene il carattere c;
- $\forall q \in Q$, il predicato $H_q(\vec{p}, \vec{t})$ indica che, al tempo $\vec{t}$, N si trova nello stato q e la testina si trova in $\vec{p}$;
- $\Psi \in FO(L \cup \{<, T_0, T_1, T_\sqcup\} \cup \{H_q | q \in Q\})$.

Dimostrazione (2)

Costruiamo la formula $\Phi \in \exists SO$:

$$\Phi = \exists < \exists T_0 \exists T_1 \exists T_\sqcup \exists H_{q_0} .. \exists H_{q_m} \Psi$$

dove

- $<$ è un ordine lineare;
- $\forall c \in \Delta$, il predicato $T_c(\vec{p}, \vec{t})$ indica che la posizione $\vec{p}$ del nastro, al tempo $\vec{t}$, contiene il carattere c;
- $\forall q \in Q$, il predicato $H_q(\vec{p}, \vec{t})$ indica che, al tempo $\vec{t}$, N si trova nello stato q e la testina si trova in $\vec{p}$;
- $\Psi \in FO(L \cup \{<, T_0, T_1, T_\sqcup\} \cup \{H_q | q \in Q\})$.

Dimostrazione (2)

Costruiamo la formula $\Phi \in \exists SO$:

$$\Phi = \exists < \exists T_0 \exists T_1 \exists T_\sqcup \exists H_{q_0} .. \exists H_{q_m} \Psi$$

dove

- $<$ è un ordine lineare;
- $\forall c \in \Delta$, il predicato $T_c(\vec{p}, \vec{t})$ indica che la posizione $\vec{p}$ del nastro, al tempo $\vec{t}$, contiene il carattere c;
- $\forall q \in Q$, il predicato $H_q(\vec{p}, \vec{t})$ indica che, al tempo $\vec{t}$, N si trova nello stato q e la testina si trova in $\vec{p}$;
- $\Psi \in FO(L \cup \{<, T_0, T_1, T_\sqcup\} \cup \{H_q | q \in Q\})$.

Dimostrazione (2)

Costruiamo la formula $\Phi \in \exists SO$:

$$\Phi = \exists < \exists T_0 \exists T_1 \exists T_\sqcup \exists H_{q_0} .. \exists H_{q_m} \Psi$$

dove

- $<$ è un ordine lineare;
- $\forall c \in \Delta$, il predicato $T_c(\vec{p}, \vec{t})$ indica che la posizione $\vec{p}$ del nastro, al tempo $\vec{t}$, contiene il carattere c;
- $\forall q \in Q$, il predicato $H_q(\vec{p}, \vec{t})$ indica che, al tempo $\vec{t}$, N si trova nello stato q e la testina si trova in $\vec{p}$;
- $\Psi \in FO(L \cup \{<, T_0, T_1, T_\sqcup\} \cup \{H_q | q \in Q\})$.

Dimostrazione (2)

Ψ è creato dalla congiunzione delle seguenti affermazioni:

1. $<$ definisce un ordine lineare;
2. dato un istante di tempo $\vec{t}$:
   - ogni cella del nastro $\vec{p}$ contiene esattamente un elemento di Δ
   - N si trova in esattamente uno stato di Q
3. la computazione di N raggiunge uno stato di accettazione;
4. i predicati T_i e H_q rispettano la funzione di transizione δ:

$$\bigvee_{(q',b,m)\in\delta(q,a)} \alpha_{(q,a,q',b,m)}$$

   dove α descrive la transizione:
   - se N si trova nello stato q e la testina si trova nel carattere a
   - sostituisce il carattere nella posizione della testina con il carattere b
   - sposta la testina secondo m
   - cambia lo stato in q'
5. la computazione inizia dalla configurazione iniziale (richiede L).

Dimostrazione (2)

Ψ è creato dalla congiunzione delle seguenti affermazioni:

1. $<$ definisce un ordine lineare;
2. dato un istante di tempo $\vec{t}$:
   - ogni cella del nastro $\vec{p}$ contiene esattamente un elemento di Δ
   - N si trova in esattamente uno stato di Q
3. la computazione di N raggiunge uno stato di accettazione;
4. i predicati T_i e H_q rispettano la funzione di transizione δ:

$$\bigvee_{(q',b,m)\in\delta(q,a)} \alpha_{(q,a,q',b,m)}$$

   dove α descrive la transizione:
   - se N si trova nello stato q e la testina si trova nel carattere a
   - sostituisce il carattere nella posizione della testina con il carattere b
   - sposta la testina secondo m
   - cambia lo stato in q'
5. la computazione inizia dalla configurazione iniziale (richiede L).

Dimostrazione (2)

Ψ è creato dalla congiunzione delle seguenti affermazioni:

1. $<$ definisce un ordine lineare;
2. dato un istante di tempo $\vec{t}$:
   - ogni cella del nastro $\vec{p}$ contiene esattamente un elemento di Δ
   - N si trova in esattamente uno stato di Q
3. la computazione di N raggiunge uno stato di accettazione;
4. i predicati T_i e H_q rispettano la funzione di transizione δ:

$$\bigvee_{(q',b,m)\in\delta(q,a)} \alpha_{(q,a,q',b,m)}$$

   dove α descrive la transizione:
   - se N si trova nello stato q e la testina si trova nel carattere a
   - sostituisce il carattere nella posizione della testina con il carattere b
   - sposta la testina secondo m
   - cambia lo stato in q'
5. la computazione inizia dalla configurazione iniziale (richiede L).

Dimostrazione (2)

Ψ è creato dalla congiunzione delle seguenti affermazioni:

1. $<$ definisce un ordine lineare;
2. dato un istante di tempo $\vec{t}$:
   - ogni cella del nastro $\vec{p}$ contiene esattamente un elemento di Δ
   - N si trova in esattamente uno stato di Q
3. la computazione di N raggiunge uno stato di accettazione;
4. i predicati T_i e H_q rispettano la funzione di transizione δ:

$$\bigvee_{(q',b,m)\in\delta(q,a)} \alpha_{(q,a,q',b,m)}$$

   dove α descrive la transizione:
   - se N si trova nello stato q e la testina si trova nel carattere a
   - sostituisce il carattere nella posizione della testina con il carattere b
   - sposta la testina secondo m
   - cambia lo stato in q'
5. la computazione inizia dalla configurazione iniziale (richiede L).

Dimostrazione (2)

Ψ è creato dalla congiunzione delle seguenti affermazioni:

1. $<$ definisce un ordine lineare;
2. dato un istante di tempo $\vec{t}$:
   - ogni cella del nastro $\vec{p}$ contiene esattamente un elemento di Δ
   - N si trova in esattamente uno stato di Q
3. la computazione di N raggiunge uno stato di accettazione;
4. i predicati T_i e H_q rispettano la funzione di transizione δ:

$$\bigvee_{(q',b,m)\in\delta(q,a)} \alpha_{(q,a,q',b,m)}$$

   dove α descrive la transizione:
   - se N si trova nello stato q e la testina si trova nel carattere a
   - sostituisce il carattere nella posizione della testina con il carattere b
   - sposta la testina secondo m
   - cambia lo stato in q'
5. la computazione inizia dalla configurazione iniziale (richiede L).

Dimostrazione (2)

5. la computazione inizia dalla configurazione iniziale (richiede L).

Supponiamo di avere due formule, $\psi_1, \psi_2 \in FO(L \cup \{<\})$ con il seguente significato:

$\mathcal{I} \models \psi_1(\vec{p}) \iff$ la posizione $\vec{p}$ della codifica di $\mathcal{I}$ contiene il carattere 1;
$\mathcal{I} \models \psi_2(\vec{p}) \iff$ la posizione $\vec{p}$ è maggiore della lunghezza della codifica di $\mathcal{I}$;

dove $\psi_1(\vec{p})$ e $\psi_2(\vec{p})$ dipendono esclusivamente dal linguaggio e non dalla struttura $\mathcal{I}$.

Costruiamo la configurazione iniziale con la formula:

$$\forall\vec{p}\forall\vec{t}(\neg\exists\vec{u}(\vec{u} <_k \vec{t}) \implies [(\psi_1(\vec{p}) \iff T_1(\vec{p},\vec{t})) \wedge (\psi_2(\vec{p}) \iff T_\sqcup(\vec{p},\vec{t}))])$$
$$\wedge$$
$$\forall\vec{p}\forall\vec{t}((\neg\exists\vec{u}(\vec{u} <_k \vec{t}) \wedge \neg\exists\vec{u}(\vec{u} <_k \vec{p})) \implies H_{q_0}(\vec{p},\vec{t}))$$

Dimostrazione (2)

5. la computazione inizia dalla configurazione iniziale (richiede L).

Supponiamo di avere due formule, $\psi_1, \psi_2 \in FO(L \cup \{<\})$ con il seguente significato:

$\mathcal{I} \models \psi_1(\vec{p}) \iff$ la posizione $\vec{p}$ della codifica di $\mathcal{I}$ contiene il carattere 1;
$\mathcal{I} \models \psi_2(\vec{p}) \iff$ la posizione $\vec{p}$ è maggiore della lunghezza della codifica di $\mathcal{I}$;

dove $\psi_1(\vec{p})$ e $\psi_2(\vec{p})$ dipendono esclusivamente dal linguaggio e non dalla struttura $\mathcal{I}$.

Costruiamo la configurazione iniziale con la formula:

$$\forall\vec{p}\forall\vec{t}(\neg\exists\vec{u}(\vec{u} <_k \vec{t}) \implies [(\psi_1(\vec{p}) \iff T_1(\vec{p},\vec{t})) \wedge (\psi_2(\vec{p}) \iff T_\sqcup(\vec{p},\vec{t}))])$$
$$\wedge$$
$$\forall\vec{p}\forall\vec{t}((\neg\exists\vec{u}(\vec{u} <_k \vec{t}) \wedge \neg\exists\vec{u}(\vec{u} <_k \vec{p})) \implies H_{q_0}(\vec{p},\vec{t}))$$

Dimostrazione (2)

5. la computazione inizia dalla configurazione iniziale (richiede L).

Supponiamo di avere due formule, $\psi_1, \psi_2 \in FO(L \cup \{<\})$ con il seguente significato:

$\mathcal{I} \models \psi_1(\vec{p}) \iff$ la posizione $\vec{p}$ della codifica di $\mathcal{I}$ contiene il carattere 1;
$\mathcal{I} \models \psi_2(\vec{p}) \iff$ la posizione $\vec{p}$ è maggiore della lunghezza della codifica di $\mathcal{I}$;

dove $\psi_1(\vec{p})$ e $\psi_2(\vec{p})$ dipendono esclusivamente dal linguaggio e non dalla struttura $\mathcal{I}$.

Costruiamo la configurazione iniziale con la formula:

$$\forall\vec{p}\forall\vec{t}(\neg\exists\vec{u}(\vec{u} <_k \vec{t}) \implies [(\psi_1(\vec{p}) \iff T_1(\vec{p},\vec{t})) \wedge (\psi_2(\vec{p}) \iff T_\sqcup(\vec{p},\vec{t}))])$$
$$\wedge$$
$$\forall\vec{p}\forall\vec{t}((\neg\exists\vec{u}(\vec{u} <_k \vec{t}) \wedge \neg\exists\vec{u}(\vec{u} <_k \vec{p})) \implies H_{q_0}(\vec{p},\vec{t}))$$

Esempio di costruzione di ψ_1 nel caso di grafi

Per $k = 3$, $\vec{p}$ è la posizione $p_1 \cdot n^2 + p_2 \cdot n + p_3$ nell'ordine lessicografico delle k-puple. Per semplicità consideriamo il minimo nell'ordine definito da $<$ come 0, e il suo successore come 1.

La formula $\psi_1(p_1, p_2, p_3)$ deve esprimere che:

- se $p_1 > 1$ allora ψ_1 è falso (la codifica di E termina in posizione $n^2 + n$), assumiamo $p_1 = 0$;
- la posizione n contiene il bit successivo a 0^n, quindi se $p_2 = 0$ allora ψ_1 è falso, quindi $p_2 \neq 0$;
- se $p_3 \neq 0$

$$p_2 \cdot n + p_3 = (p_2 - 1) \cdot n + (p_3 - 1) + (n + 1)$$

  che corrisponde all'arco $E(p_2 - 1, p_3 - 1)$;
- se $p_3 = 0$

$$p_2 \cdot n = (p_2 - 2) \cdot n + (n - 1) + (n + 1)$$

  che corrisponde all'arco $E(p_2 - 2, n - 1)$.

Analogo per $p_1 = 1$.

Esempio di costruzione di ψ_1 nel caso di grafi

Per $k = 3$, $\vec{p}$ è la posizione $p_1 \cdot n^2 + p_2 \cdot n + p_3$ nell'ordine lessicografico delle k-puple. Per semplicità consideriamo il minimo nell'ordine definito da $<$ come 0, e il suo successore come 1.

La formula $\psi_1(p_1, p_2, p_3)$ deve esprimere che:

- se $p_1 > 1$ allora ψ_1 è falso (la codifica di E termina in posizione $n^2 + n$), assumiamo $p_1 = 0$;
- la posizione n contiene il bit successivo a 0^n, quindi se $p_2 = 0$ allora ψ_1 è falso, quindi $p_2 \neq 0$;
- se $p_3 \neq 0$

$$p_2 \cdot n + p_3 = (p_2 - 1) \cdot n + (p_3 - 1) + (n + 1)$$

che corrisponde all'arco $E(p_2 - 1, p_3 - 1)$;
- se $p_3 = 0$

$$p_2 \cdot n = (p_2 - 2) \cdot n + (n - 1) + (n + 1)$$

che corrisponde all'arco $E(p_2 - 2, n - 1)$.

Analogo per $p_1 = 1$.

Esempio di costruzione di ψ_1 nel caso di grafi

Per $k = 3$, $\vec{p}$ è la posizione $p_1 \cdot n^2 + p_2 \cdot n + p_3$ nell'ordine lessicografico delle k-puple. Per semplicità consideriamo il minimo nell'ordine definito da $<$ come 0, e il suo successore come 1.

La formula $\psi_1(p_1, p_2, p_3)$ deve esprimere che:

- se $p_1 > 1$ allora ψ_1 è falso (la codifica di E termina in posizione $n^2 + n$), assumiamo $p_1 = 0$;
- la posizione n contiene il bit successivo a 0^n, quindi se $p_2 = 0$ allora ψ_1 è falso, quindi $p_2 \neq 0$;
- se $p_3 \neq 0$
$$p_2 \cdot n + p_3 = (p_2 - 1) \cdot n + (p_3 - 1) + (n + 1)$$
che corrisponde all'arco $E(p_2 - 1, p_3 - 1)$;
- se $p_3 = 0$
$$p_2 \cdot n = (p_2 - 2) \cdot n + (n - 1) + (n + 1)$$
che corrisponde all'arco $E(p_2 - 2, n - 1)$.

Analogo per $p_1 = 1$.

Esempio di costruzione di ψ_1 nel caso di grafi

Per $k = 3$, $\vec{p}$ è la posizione $p_1 \cdot n^2 + p_2 \cdot n + p_3$ nell'ordine lessicografico delle k-puple. Per semplicità consideriamo il minimo nell'ordine definito da $<$ come 0, e il suo successore come 1.

La formula $\psi_1(p_1, p_2, p_3)$ deve esprimere che:

- se $p_1 > 1$ allora ψ_1 è falso (la codifica di E termina in posizione $n^2 + n$), assumiamo $p_1 = 0$;
- la posizione n contiene il bit successivo a 0^n, quindi se $p_2 = 0$ allora ψ_1 è falso, quindi $p_2 \neq 0$;
- se $p_3 \neq 0$
$$p_2 \cdot n + p_3 = (p_2 - 1) \cdot n + (p_3 - 1) + (n + 1)$$
che corrisponde all'arco $E(p_2 - 1, p_3 - 1)$;
- se $p_3 = 0$
$$p_2 \cdot n = (p_2 - 2) \cdot n + (n - 1) + (n + 1)$$
che corrisponde all'arco $E(p_2 - 2, n - 1)$.

Analogo per $p_1 = 1$.

Esempio di costruzione di ψ_1 nel caso di grafi

Per $k = 3$, $\vec{p}$ è la posizione $p_1 \cdot n^2 + p_2 \cdot n + p_3$ nell'ordine lessicografico delle k-puple. Per semplicità consideriamo il minimo nell'ordine definito da $<$ come 0, e il suo successore come 1.

La formula $\psi_1(p_1, p_2, p_3)$ deve esprimere che:

- se $p_1 > 1$ allora ψ_1 è falso (la codifica di E termina in posizione $n^2 + n$), assumiamo $p_1 = 0$;
- la posizione n contiene il bit successivo a 0^n, quindi se $p_2 = 0$ allora ψ_1 è falso, quindi $p_2 \neq 0$;
- se $p_3 \neq 0$
$$p_2 \cdot n + p_3 = (p_2 - 1) \cdot n + (p_3 - 1) + (n + 1)$$
che corrisponde all'arco $E(p_2 - 1, p_3 - 1)$;
- se $p_3 = 0$
$$p_2 \cdot n = (p_2 - 2) \cdot n + (n - 1) + (n + 1)$$
che corrisponde all'arco $E(p_2 - 2, n - 1)$.

Analogo per $p_1 = 1$.

Esempio di costruzione di ψ_1 e ψ_2 nel caso di grafi

Quindi $\psi_1(\vec{p})$ e $\psi_2(\vec{p})$ sono esprimibili tramite le formule

$$\begin{aligned}
\psi_1(p_1, p_2, p_3) = & \\
& [p_1 = 0 \wedge p_2 = 1 \wedge p_3 = 0] \\
& \vee [(p_1 = 0 \wedge p_2 \neq 0 \wedge p_3 = 0) \wedge E(p_2 - 2, n - 1)] \\
& \vee [(p_1 = 0 \wedge p_2 \neq 0 \wedge p_3 \neq 0) \wedge E(p_2 - 1, p_3 - 1)] \\
& \vee [(p_1 = 1 \wedge p_2 = 0 \wedge p_3 = 0) \wedge E(n - 2, n - 1)] \\
& \vee [(p_1 = 1 \wedge p_2 = 0 \wedge p_3 \neq 0) \wedge E(n - 2, p_3 - 1)] \\
& \vee [(p_1 = 1 \wedge p_2 = 1 \wedge p_3 = 0) \wedge E(n - 1, n - 1)] \\
\psi_2(p_1, p_2, p_3) = & \\
& (p_1 = 1 \implies ((p_2 > 1) \vee (p_2 > 0 \wedge p_3 > 0)) \\
& \vee (p_1 > 1)
\end{aligned}$$

□

Osservazioni

Mentre nel caso generale e nel caso di restrizione di SO a MSO è noto un esempio che dimostra, rispettivamente, $\exists SO \neq \forall SO$ e $\exists MSO \neq \forall MSO$, $\exists SO \stackrel{?}{=} \forall SO$ rimane un problema aperto nella teoria dei modelli finiti.

(Si ricorda inoltre che $coNP = \{L : \bar{L} \in NP\}$)

- Dimostrando che $\exists SO$ cattura NP otteniamo che coNP è catturato dalla negazione di $\exists SO$, ovvero $\forall SO$;
- Dimostrare $\exists SO \neq \forall SO$ equivale quindi a dimostrare $NP \neq coNP$;
- Dimostrare $\exists SO \neq \forall SO$ implicherebbe inoltre $P \neq NP$, poichè $P = coP$;
- Dimostrare $\exists SO = \forall SO$ implicherebbe $\Sigma_i^P = \Pi_i^P$, e $PH = NP$.

Osservazioni

Mentre nel caso generale e nel caso di restrizione di SO a MSO è noto un esempio che dimostra, rispettivamente, $\exists SO \neq \forall SO$ e $\exists MSO \neq \forall MSO$, $\exists SO \stackrel{?}{=} \forall SO$ rimane un problema aperto nella teoria dei modelli finiti.

(Si ricorda inoltre che $coNP = \{L : \bar{L} \in NP\}$)

- Dimostrando che $\exists SO$ cattura NP otteniamo che coNP è catturato dalla negazione di $\exists SO$, ovvero $\forall SO$;
- Dimostrare $\exists SO \neq \forall SO$ equivale quindi a dimostrare $NP \neq coNP$;
- Dimostrare $\exists SO \neq \forall SO$ implicherebbe inoltre $P \neq NP$, poichè $P = coP$;
- Dimostrare $\exists SO = \forall SO$ implicherebbe $\Sigma_i^P = \Pi_i^P$, e $PH = NP$.

Osservazioni

Mentre nel caso generale e nel caso di restrizione di SO a MSO è noto un esempio che dimostra, rispettivamente, $\exists SO \neq \forall SO$ e $\exists MSO \neq \forall MSO$, $\exists SO \stackrel{?}{=} \forall SO$ rimane un problema aperto nella teoria dei modelli finiti.

(Si ricorda inoltre che $coNP = \{L : \bar{L} \in NP\}$)

- Dimostrando che $\exists SO$ cattura NP otteniamo che coNP è catturato dalla negazione di $\exists SO$, ovvero $\forall SO$;
- Dimostrare $\exists SO \neq \forall SO$ equivale quindi a dimostrare $NP \neq coNP$;
- Dimostrare $\exists SO \neq \forall SO$ implicherebbe inoltre $P \neq NP$, poichè $P = coP$;
- Dimostrare $\exists SO = \forall SO$ implicherebbe $\Sigma_i^P = \Pi_i^P$, e $PH = NP$.

Osservazioni

Mentre nel caso generale e nel caso di restrizione di SO a MSO è noto un esempio che dimostra, rispettivamente, $\exists SO \neq \forall SO$ e $\exists MSO \neq \forall MSO$, $\exists SO \stackrel{?}{=} \forall SO$ rimane un problema aperto nella teoria dei modelli finiti.

(Si ricorda inoltre che $coNP = \{L : \bar{L} \in NP\}$)

- Dimostrando che $\exists SO$ cattura NP otteniamo che coNP è catturato dalla negazione di $\exists SO$, ovvero $\forall SO$;
- Dimostrare $\exists SO \neq \forall SO$ equivale quindi a dimostrare $NP \neq coNP$;
- Dimostrare $\exists SO \neq \forall SO$ implicherebbe inoltre $P \neq NP$, poichè $P = coP$;
- Dimostrare $\exists SO = \forall SO$ implicherebbe $\Sigma_i^P = \Pi_i^P$, e $PH = NP$.

Osservazioni

Mentre nel caso generale e nel caso di restrizione di SO a MSO è noto un esempio che dimostra, rispettivamente, $\exists SO \neq \forall SO$ e $\exists MSO \neq \forall MSO$, $\exists SO \stackrel{?}{=} \forall SO$ rimane un problema aperto nella teoria dei modelli finiti.

(Si ricorda inoltre che $coNP = \{L : \bar{L} \in NP\}$)

- Dimostrando che $\exists SO$ cattura NP otteniamo che coNP è catturato dalla negazione di $\exists SO$, ovvero $\forall SO$;
- Dimostrare $\exists SO \neq \forall SO$ equivale quindi a dimostrare $NP \neq coNP$;
- Dimostrare $\exists SO \neq \forall SO$ implicherebbe inoltre $P \neq NP$, poichè $P = coP$;
- Dimostrare $\exists SO = \forall SO$ implicherebbe $\Sigma_i^P = \Pi_i^P$, e $PH = NP$.

Differenze con Trakhtenbrot

Trakhtenbrot	Fagin
Nessun limite a tempo di esecuzione e spazio utilizzato	Tempo e spazio limitati da un polinomio del dominio
T_i e H_q rispettano la funzione di transizione	T_i e H_q rispettano una delle possibili transizioni
La macchina inizia su input vuoto	La configurazione iniziale contiene la codifica della struttura
La computazione termina	La computazione raggiunge uno stato di accettazione
Indipendenza dal linguaggio	Dipendenza dal linguaggio nella formulazione di ψ_1 e ψ_2

Differenze con Trakhtenbrot

Trakhtenbrot	Fagin
Nessun limite a tempo di esecuzione e spazio utilizzato	Tempo e spazio limitati da un polinomio del dominio
T_i e H_q rispettano la funzione di transizione	T_i e H_q rispettano una delle possibili transizioni
La macchina inizia su input vuoto	La configurazione iniziale contiene la codifica della struttura
La computazione termina	La computazione raggiunge uno stato di accettazione
Indipendenza dal linguaggio	Dipendenza dal linguaggio nella formulazione di ψ_1 e ψ_2

Differenze con Trakhtenbrot

Trakhtenbrot	Fagin
Nessun limite a tempo di esecuzione e spazio utilizzato	Tempo e spazio limitati da un polinomio del dominio
T_i e H_q rispettano la funzione di transizione	T_i e H_q rispettano una delle possibili transizioni
La macchina inizia su input vuoto	La configurazione iniziale contiene la codifica della struttura
La computazione termina	La computazione raggiunge uno stato di accettazione
Indipendenza dal linguaggio	Dipendenza dal linguaggio nella formulazione di ψ_1 e ψ_2

Differenze con Trakhtenbrot

Trakhtenbrot	Fagin
Nessun limite a tempo di esecuzione e spazio utilizzato	Tempo e spazio limitati da un polinomio del dominio
T_i e H_q rispettano la funzione di transizione	T_i e H_q rispettano una delle possibili transizioni
La macchina inizia su input vuoto	La configurazione iniziale contiene la codifica della struttura
La computazione termina	La computazione raggiunge uno stato di accettazione
Indipendenza dal linguaggio	Dipendenza dal linguaggio nella formulazione di ψ_1 e ψ_2

Differenze con Trakhtenbrot

Trakhtenbrot	Fagin
Nessun limite a tempo di esecuzione e spazio utilizzato	Tempo e spazio limitati da un polinomio del dominio
T_i e H_q rispettano la funzione di transizione	T_i e H_q rispettano una delle possibili transizioni
La macchina inizia su input vuoto	La configurazione iniziale contiene la codifica della struttura
La computazione termina	La computazione raggiunge uno stato di accettazione
Indipendenza dal linguaggio	Dipendenza dal linguaggio nella formulazione di ψ_1 e ψ_2

Differenze con Trakhtenbrot

Trakhtenbrot	Fagin
Nessun limite a tempo di esecuzione e spazio utilizzato	Tempo e spazio limitati da un polinomio del dominio
T_i e H_q rispettano la funzione di transizione	T_i e H_q rispettano una delle possibili transizioni
La macchina inizia su input vuoto	La configurazione iniziale contiene la codifica della struttura
La computazione termina	La computazione raggiunge uno stato di accettazione
Indipendenza dal linguaggio	Dipendenza dal linguaggio nella formulazione di ψ_1 e ψ_2

Differenze con Trakhtenbrot

Trakhtenbrot	Fagin
Nessun limite a tempo di esecuzione e spazio utilizzato	Tempo e spazio limitati da un polinomio del dominio
T_i e H_q rispettano la funzione di transizione	T_i e H_q rispettano una delle possibili transizioni
La macchina inizia su input vuoto	La configurazione iniziale contiene la codifica della struttura
La computazione termina	La computazione raggiunge uno stato di accettazione
Indipendenza dal linguaggio	Dipendenza dal linguaggio nella formulazione di ψ_1 e ψ_2

Teorema di Cook

Teorema di Cook

Teorema di Cook

SAT è NP-completo

Dimostrazione

Sia P un problema NP ed N la NMdT che accetta strutture che soddisfano P.

Dal teorema di Fagin sappiamo che esiste una formula $\Phi = \exists S_1, .., \exists S_n \varphi$ tale che N accetta $\mathcal{I} \iff \mathcal{I} \models \Phi$.

Sia $X = \{S_i(\vec{a}) : i \in \{1, .., n\}, \vec{a} \in A^{arity(S_i)}\}$ l'insieme delle scelte non deterministiche.
Costruiamo la formula $\alpha_\varphi^{\mathcal{I}}$ modificando φ nel seguente modo:

- ogni $\exists x \psi(x, \cdot)$ diventa $\bigvee_{a \in A} \psi(a, \cdot)$;
- ogni $\forall x \psi(x, \cdot)$ diventa $\bigwedge_{a \in A} \psi(a, \cdot)$;
- ogni $R(\vec{a})$, per $R \in L$, diventa il suo valore di verità in $\mathcal{I}$.

Le variabili della formula $\alpha_\varphi^{\mathcal{I}}$ sono definite in X.

Abbiamo $\mathcal{I} \models \Phi \iff \alpha_\varphi^{\mathcal{I}}$ è soddisfacibile.

La formula $\alpha_\varphi^{\mathcal{I}}$ può essere costruita in spazio logaritmico. □

Dimostrazione

Sia P un problema NP ed N la NMdT che accetta strutture che soddisfano P.

Dal teorema di Fagin sappiamo che esiste una formula $\Phi = \exists S_1, .., \exists S_n \varphi$ tale che N accetta $\mathcal{I} \iff \mathcal{I} \models \Phi$.

Sia $X = \{S_i(\vec{a}) : i \in \{1, .., n\}, \vec{a} \in A^{arity(S_i)}\}$ l'insieme delle scelte non deterministiche.
Costruiamo la formula $\alpha_\varphi^{\mathcal{I}}$ modificando φ nel seguente modo:

- ogni $\exists x \psi(x, \cdot)$ diventa $\bigvee_{a \in A} \psi(a, \cdot)$;
- ogni $\forall x \psi(x, \cdot)$ diventa $\bigwedge_{a \in A} \psi(a, \cdot)$;
- ogni $R(\vec{a})$, per $R \in L$, diventa il suo valore di verità in $\mathcal{I}$.

Le variabili della formula $\alpha_\varphi^{\mathcal{I}}$ sono definite in X.

Abbiamo $\mathcal{I} \models \Phi \iff \alpha_\varphi^{\mathcal{I}}$ è soddisfacibile.

La formula $\alpha_\varphi^{\mathcal{I}}$ può essere costruita in spazio logaritmico. □

Dimostrazione

Sia P un problema NP ed N la NMdT che accetta strutture che soddisfano P.

Dal teorema di Fagin sappiamo che esiste una formula $\Phi = \exists S_1, .., \exists S_n \varphi$ tale che N accetta $\mathcal{I} \iff \mathcal{I} \models \Phi$.

Sia $X = \{S_i(\vec{a}) : i \in \{1, .., n\}, \vec{a} \in A^{arity(S_i)}\}$ l'insieme delle scelte non deterministiche.
Costruiamo la formula $\alpha_\varphi^{\mathcal{I}}$ modificando φ nel seguente modo:

- ogni $\exists x \psi(x, \cdot)$ diventa $\bigvee_{a \in A} \psi(a, \cdot)$;
- ogni $\forall x \psi(x, \cdot)$ diventa $\bigwedge_{a \in A} \psi(a, \cdot)$;
- ogni $R(\vec{a})$, per $R \in L$, diventa il suo valore di verità in $\mathcal{I}$.

Le variabili della formula $\alpha_\varphi^{\mathcal{I}}$ sono definite in X.

Abbiamo $\mathcal{I} \models \Phi \iff \alpha_\varphi^{\mathcal{I}}$ è soddisfacibile.

La formula $\alpha_\varphi^{\mathcal{I}}$ può essere costruita in spazio logaritmico. □

Dimostrazione

Sia P un problema NP ed N la NMdT che accetta strutture che soddisfano P.

Dal teorema di Fagin sappiamo che esiste una formula $\Phi = \exists S_1, .., \exists S_n \varphi$ tale che N accetta $\mathcal{I} \iff \mathcal{I} \models \Phi$.

Sia $X = \{S_i(\vec{a}) : i \in \{1, .., n\}, \vec{a} \in A^{arity(S_i)}\}$ l'insieme delle scelte non deterministiche.
Costruiamo la formula $\alpha_\varphi^{\mathcal{I}}$ modificando φ nel seguente modo:

- ogni $\exists x \psi(x, \cdot)$ diventa $\bigvee_{a \in A} \psi(a, \cdot)$;
- ogni $\forall x \psi(x, \cdot)$ diventa $\bigwedge_{a \in A} \psi(a, \cdot)$;
- ogni $R(\vec{a})$, per $R \in L$, diventa il suo valore di verità in $\mathcal{I}$.

Le variabili della formula $\alpha_\varphi^{\mathcal{I}}$ sono definite in X.

Abbiamo $\mathcal{I} \models \Phi \iff \alpha_\varphi^{\mathcal{I}}$ è soddisfacibile.

La formula $\alpha_\varphi^{\mathcal{I}}$ può essere costruita in spazio logaritmico. □

Dimostrazione

Sia P un problema NP ed N la NMdT che accetta strutture che soddisfano P.

Dal teorema di Fagin sappiamo che esiste una formula $\Phi = \exists S_1, .., \exists S_n \varphi$ tale che N accetta $\mathcal{I} \iff \mathcal{I} \models \Phi$.

Sia $X = \{S_i(\vec{a}) : i \in \{1, .., n\}, \vec{a} \in A^{arity(S_i)}\}$ l'insieme delle scelte non deterministiche.
Costruiamo la formula $\alpha_\varphi^{\mathcal{I}}$ modificando φ nel seguente modo:

- ogni $\exists x \psi(x, \cdot)$ diventa $\bigvee_{a \in A} \psi(a, \cdot)$;
- ogni $\forall x \psi(x, \cdot)$ diventa $\bigwedge_{a \in A} \psi(a, \cdot)$;
- ogni $R(\vec{a})$, per $R \in L$, diventa il suo valore di verità in $\mathcal{I}$.

Le variabili della formula $\alpha_\varphi^{\mathcal{I}}$ sono definite in X.

Abbiamo $\mathcal{I} \models \Phi \iff \alpha_\varphi^{\mathcal{I}}$ è soddisfacibile.

La formula $\alpha_\varphi^{\mathcal{I}}$ può essere costruita in spazio logaritmico. □

Dimostrazione

Sia P un problema NP ed N la NMdT che accetta strutture che soddisfano P.

Dal teorema di Fagin sappiamo che esiste una formula $\Phi = \exists S_1, .., \exists S_n \varphi$ tale che N accetta $\mathcal{I} \iff \mathcal{I} \models \Phi$.

Sia $X = \{S_i(\vec{a}) : i \in \{1, .., n\}, \vec{a} \in A^{arity(S_i)}\}$ l'insieme delle scelte non deterministiche.
Costruiamo la formula $\alpha_\varphi^{\mathcal{I}}$ modificando φ nel seguente modo:

- ogni $\exists x \psi(x, \cdot)$ diventa $\bigvee_{a \in A} \psi(a, \cdot)$;
- ogni $\forall x \psi(x, \cdot)$ diventa $\bigwedge_{a \in A} \psi(a, \cdot)$;
- ogni $R(\vec{a})$, per $R \in L$, diventa il suo valore di verità in $\mathcal{I}$.

Le variabili della formula $\alpha_\varphi^{\mathcal{I}}$ sono definite in X.

Abbiamo $\mathcal{I} \models \Phi \iff \alpha_\varphi^{\mathcal{I}}$ è soddisfacibile.

La formula $\alpha_\varphi^{\mathcal{I}}$ può essere costruita in spazio logaritmico. □

Gerarchia Polinomiale

Gerarchia polinomiale e gerarchia analitica

La gerarchia polinomiale è definita come:

$$\Delta_0^P = \Sigma_0^P = \Pi_0^P = P$$
$$\Sigma_1^P = NP \text{ , } \Pi_1^P = coNP$$
$$\Sigma_{n+1}^P = NP^{\Sigma_n^P}$$
$$\Pi_{n+1}^P = coNP^{\Sigma_n^P}$$
$$\Delta_{n+1}^P = P^{\Sigma_n^P}$$

La gerarchia analitica è definita come:

$$\Sigma_0^1 = \Pi_0^1 = \text{SO senza quantificatori sulle relazioni}$$
$$\Sigma_{n+1}^1 = \exists X_1..\exists X_k\psi, \text{ dove } \psi \in \Pi_n^1$$
$$\Pi_{n+1}^1 = \forall X_1..\forall X_k\psi, \text{ dove } \psi \in \Sigma_n^1$$

Σ_k^1 cattura Σ_k^P e Π_k^1 cattura Π_k^P

Il teorema di Fagin ci dice che

- $\exists SO$ cattura NP (Σ_1^1 cattura Σ_1^P)
- $\forall SO$ cattura coNP (Π_1^1 cattura Π_1^P)

Supponiamo di avere $P \in NP^{\Sigma_k^P}$, quindi decidibile tramite una MdT non deterministica in tempo polinomiale con oracoli in Σ_k^P.

Per il teorema di Fagin esiste $\Phi \in \exists SO$ con dei predicati per esprimere formule Σ_k^P, corrispondenti all'oracolo.
Per ipotesi induttiva, sappiamo che i predicati per Σ_k^P sono esprimibili in Σ_k^1

Spostando i quantificatori del secondo ordine all'inizio, trasformiamo Ψ in una formula Σ_{k+1}^1.

Σ_k^1 cattura Σ_k^P e Π_k^1 cattura Π_k^P

Il teorema di Fagin ci dice che

- $\exists SO$ cattura NP (Σ_1^1 cattura Σ_1^P)
- $\forall SO$ cattura coNP (Π_1^1 cattura Π_1^P)

Supponiamo di avere $P \in NP^{\Sigma_k^P}$, quindi decidibile tramite una MdT non deterministica in tempo polinomiale con oracoli in Σ_k^P.

Per il teorema di Fagin esiste $\Phi \in \exists SO$ con dei predicati per esprimere formule Σ_k^P, corrispondenti all'oracolo.

Per ipotesi induttiva, sappiamo che i predicati per Σ_k^P sono esprimibili in Σ_k^1

Spostando i quantificatori del secondo ordine all'inizio, trasformiamo Ψ in una formula Σ_{k+1}^1.

Σ_k^1 cattura Σ_k^P e Π_k^1 cattura Π_k^P

Il teorema di Fagin ci dice che

- $\exists SO$ cattura NP (Σ_1^1 cattura Σ_1^P)
- $\forall SO$ cattura coNP (Π_1^1 cattura Π_1^P)

Supponiamo di avere $P \in NP^{\Sigma_k^P}$, quindi decidibile tramite una MdT non deterministica in tempo polinomiale con oracoli in Σ_k^P.

Per il teorema di Fagin esiste $\Phi \in \exists SO$ con dei predicati per esprimere formule Σ_k^P, corrispondenti all'oracolo.
Per ipotesi induttiva, sappiamo che i predicati per Σ_k^P sono esprimibili in Σ_k^1

Spostando i quantificatori del secondo ordine all'inizio, trasformiamo Ψ in una formula Σ_{k+1}^1.

Esempio

Supponiamo di avere una proprietà P sia esprimibile in $\Sigma_2^P = NP^{\Sigma_1^P}$.

Per il teorema di Fagin sappiamo che esiste una formula $\Phi = \exists X_1..\exists X_n \varphi(P')$ dove P' è esprimibile a sua volta con una formula $\exists Y_1..\exists Y_m \psi$, $\psi \in FO$.

Se P' occorre in forma negata all'interno di Φ, mettendo Φ in forma prenessa otteniamo
$\Phi = \exists X_1..\exists X_n \neg(\exists Y_1..\exists Y_m \psi) = \exists X_1..\exists X_n(\forall Y_1..\forall Y_m \neg\psi)$
ovvero $\Phi \in \Sigma_2^1$

Altrimenti, se P' occorre in forma positiva, mettendo Φ in forma prenessa otteniamo $\Phi = \exists X_1..\exists X_n(\exists Y_1..\exists Y_m \psi) = \exists X_1..\exists X_n \exists Y_1..\exists Y_m \Psi$
ovvero $\Phi \in \Sigma_1^1$

(Simmetrico per Π) □

Esempio

Supponiamo di avere una proprietà P sia esprimibile in $\Sigma_2^P = NP^{\Sigma_1^P}$.

Per il teorema di Fagin sappiamo che esiste una formula
$\Phi = \exists X_1..\exists X_n \varphi(P')$ dove P' è esprimibile a sua volta con una formula
$\exists Y_1..\exists Y_m \psi$, $\psi \in FO$.

Se P' occorre in forma negata all'interno di Φ, mettendo Φ in forma prenessa otteniamo
$\Phi = \exists X_1..\exists X_n \neg(\exists Y_1..\exists Y_m \psi) = \exists X_1..\exists X_n(\forall Y_1..\forall Y_m \neg\psi)$
ovvero $\Phi \in \Sigma_2^1$

Altrimenti, se P' occorre in forma positiva, mettendo Φ in forma prenessa otteniamo $\Phi = \exists X_1..\exists X_n(\exists Y_1..\exists Y_m \psi) = \exists X_1..\exists X_n \exists Y_1..\exists Y_m \Psi$
ovvero $\Phi \in \Sigma_1^1$

(Simmetrico per Π) □

Esempio

Supponiamo di avere una proprietà P sia esprimibile in $\Sigma_2^P = NP^{\Sigma_1^P}$.

Per il teorema di Fagin sappiamo che esiste una formula
$\Phi = \exists X_1..\exists X_n \varphi(P')$ dove P' è esprimibile a sua volta con una formula
$\exists Y_1..\exists Y_m \psi$, $\psi \in FO$.

Se P' occorre in forma negata all'interno di Φ, mettendo Φ in forma prenessa otteniamo
$\Phi = \exists X_1..\exists X_n \neg(\exists Y_1..\exists Y_m \psi) = \exists X_1..\exists X_n(\forall Y_1..\forall Y_m \neg\psi)$
ovvero $\Phi \in \Sigma_2^1$

Altrimenti, se P' occorre in forma positiva, mettendo Φ in forma prenessa otteniamo $\Phi = \exists X_1..\exists X_n(\exists Y_1..\exists Y_m \psi) = \exists X_1..\exists X_n \exists Y_1..\exists Y_m \Psi$
ovvero $\Phi \in \Sigma_1^1$

(Simmetrico per Π) $\square$

Corollario

Abbiamo dimostrato che:

- Σ_k^1 cattura Σ_k^P;
- Π_k^1 cattura Π_k^P.

Di conseguenza, SO cattura PH.

Conclusione

- Teorema di Fagin: $\exists SO$ cattura NP, $\forall SO$ cattura coNP;
- Differenze con Trakhtenbrot;
- Teorema di Cook: SAT è NP-completo;
- SO cattura PH.

Grazie per l'attenzione.