5.

Mátrix fogalma, műveletek, determináns, rang. Speciális mátrixok, inverz. Mátrix, mint lineáris transzformáció. Sajátérték, sajátvektor.

Mátrix fogalma, műveletek, determináns, rang.

Mátrixok

Definíció

Egy m sorral és n oszloppal rendelkező számtáblázatot $m \times n$-es mátrixnak nevezünk.

$$A = \begin{pmatrix} a_{11} & a_{12} & \dots & a_{1n} \\ a_{21} & a_{22} & \dots & a_{2n} \\ \vdots & \vdots & & \vdots \\ a_{m1} & a_{m2} & \dots & a_{mn} \end{pmatrix} \qquad A \text{ elemei: } a_{ij} \qquad A = (a_{ij})$$

Az összes $m \times n$-es mátrix halmazát $\mathcal{M}_{m\times n}$-nel jelöljük.

Mátrixműveletek

1. Mátrixok összeadása: Csak azonos típusú mátrixokat tudunk összeadni. Legyenek A= (a_{ij}),B= (b_{ij}),C= (c_{ij}) m×n-es mátrixok. Ekkor C=A+B, ha $c_{ij}=a_{ij}+b_{ij}$; i= 1,...,m, j= 1,...,n.

2. Mátrixok skalárral való szorzása: Elemenként végezzük, azaz ha λ∈R, A= (a_{ij})∈$M_{m\times n}$, akkor λA= (λa_{ij})∈$M_{m\times n}$. Speciálisan: ha A és B sor-, vagy oszlopvektorok, akkor a fenti 2 művelet éppen a vektorok szokásos összeadása és skalárral való szorzása.

3. Mátrixszorzás: Legyen A= (a_{ij}) m×k, B= (b_{ij}) k×n típusú mátrix. Ekkor A és B szorzata az a C= (c_{ij}) m×n típusú mátrix, amelyre

$$c_{ij} = \sum_{r=1}^{k} a_{ir} b_{rj}.$$

A mátrixszorzás tulajdonságai:

Ha A m×n típusú, akkor $E_m \cdot A = A$ és $A \cdot E_n = A$.
Legyenek A,B mátrixok és tegyük fel, hogy létezik AB. Ha λ∈R tetszőleges, akkor λ(AB) = (λA)B=A(λB).
Ha A,B,C olyan mátrixok, hogy AB és BC létezik, akkor (AB)C=A(BC). Azaz a mátrixszorzás asszociatív. Ha A és B azonos típusú mátrixok és létezik AC, akkor BC is létezik és (A+B)C=AC+BC. Azaz teljesül a disztributivitás.
A mátrixszorzás nem kommutatív, azaz általában AB!=BA.

Legyen A egy m×n-es mátrix. Azt az A^T-vel jelölt n×m-es mátrixot, amelynek sorai az A oszlopai A transzponáltjának nevezzük.

A transzponálás tulajdonságai
$(A^T)^T=A$ (azaz a transzponálás involutív művelet)

A transzponálás és a mátrixszorzás kapcsolata: $(AB)^T=B^T \cdot A^T$.

Determinánsok:

Legyen $n \in N$ és jelölje σ az $\{1,2,...,n\}$halmaz egy permutációját, azaz legyen σ: $\{1,2,...,n\} \to \{1,2,...,n\}, i \to \sigma(i)$ bijektív függvény. (Itt σ(i) jelöli a permutációban az i. helyen álló elemet.) Azt mondjuk, hogy a σ permutációnál az i és j elem inverzióban áll, ha $i<j$ és $\sigma(i)>\sigma(j)$. Egy σ permutáció páros, ha benne az inverzióban álló párok száma páros,és páratlan, ha ez a szám páratlan.

Legyen $A = (a_{ij})$ egy $n \times n$-es kvadratikus mátrix. Az A n^2 eleméből válasszunk ki úgy n elemet, hogy minden sorból és oszlopból pontosan egyet választunk. A kiválasztott elemek alakja:

$$a_{1\sigma(1)}, a_{2\sigma(2)}, \ldots, a_{n\sigma(n)}.$$

Az A mátrix determinánsa:

$$\det(A) = |A| = \sum_{\sigma} \varepsilon(\sigma) a_{1\sigma(1)} a_{2\sigma(2)} \cdots a_{n\sigma(n)}.$$

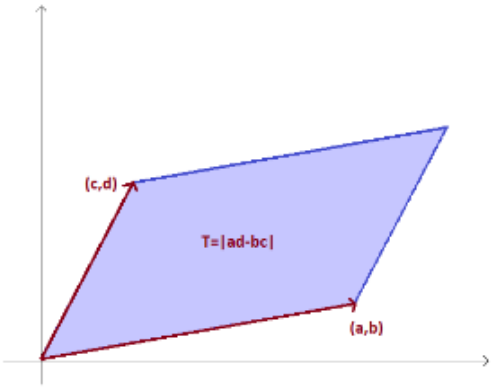
Ez az összeg $n!$ tagú. Itt: $\varepsilon(\sigma) = \begin{cases} 1, & \text{ha } \sigma \text{ páros,} \\ -1, & \text{ha } \sigma \text{ páratlan.} \end{cases}$

Ha A és B azonos rendű négyzetes mátrixok, akkor det(AB) = det(A)·det(B).

A determináns szemléletes jelentése

- másodrendű (2x2-es) determináns: a determináns sorai, mint vektorok által kifeszített parallelogramma előjeles területe

$$|A| = \begin{vmatrix} a & b \\ c & d \end{vmatrix} = ad - bc$$



- harmadrendű (3x3-as) determináns: a determináns sorai, mint vektorok által kifeszített paralelepipedon előjeles térfogata

A determináns tulajdonságai:

$\det(A) = \det(A^T)$
Ha A valamely sora csupa 0 elemből áll, akkor $\det(A) = 0$.
Ha A két sorát felcseréljük, a determináns -1-szeresére változik.
Ha A két sora egyenlő, akkor $\det(A) = 0$.
Ha A valamely sorát megszorozzuk egy λ valós számmal, akkor az így kapott mátrix determinánsa $\lambda \cdot \det(A)$.
Ha A minden sorát megszorozzuk egy λ számmal és A n-ed rendű, akkor a kapott mátrix determinánsa $\lambda^n \cdot \det(A)$.
Ha A két sora egymás skalár szorosa, akkor $\det(A) = 0$.
Egy mátrix determinánsa nem változik, ha valamely sorához hozzáadjuk egy másik sor λ-szorosát.
Ha A valamely sora előállítható a többi sor lineáris kombinációjaként, akkor $\det(A) = 0$.
A fentiek igazak sorok helyett oszlopokra is.

Ha $\det(A) \neq 0$, akkor A sorai (vagy oszlopai) lineárisan független vektorok. Ekkor ha A $n \times n$-es: sorai R^n egy bázisát alkotják.

A determináns kapcsolata az invertálással

Azt mondjuk, hogy az A négyzetes mátrix szinguláris, ha determinánsa 0. Ellenkező esetben (azaz ha $\det(A) \neq 0$)A reguláris.

Egy négyzetes mátrix pontosan akkor invertálható, ha reguláris.

Legyen A egy reguláris mátrix. Mivel $A \cdot A^{-1} = E$, ahol E az A-val azonos méretű egységmátrix, ezért a determinánsok szorzás tétele alapján $\det(A) \cdot \det(A^{-1}) = \det(E) = 1$. (Itt az, hogy $\det(E) = 1$, akár a definícióból, akár a determináns kiszámítási módjai alapján könnyen adódik.) A fenti egyenletből következik, hogy A és A^{-1} determinánsa egymás reciproka:$\det(A)^{-1} = \det(A^{-1})$.

A determináns kiszámítási módjai

1. Sarrus-szabály: 2×2-esés 3×3-as mátrixok determinánsára

2. Gauss-elimináció: bizonyos – a fenti tulajdonságokat használó –átalakítások révén a mátrixot felső háromszög alakúra hozzuk (főátló alatt csupa 0), ekkor a determináns éppen a főátlóbeli elemek szorzata. Ezek az átalakítások:
   1. sorcsere, ekkor a determináns előjelet vált;
   2. $\lambda \in R$kiemelése egy sorból;
   3. egy sorλ-szorosának hozzáadása egy másik sorhoz.

3. Kifejtési tétel: Legyen A egy n-edrendű mátrix.
   1. kiválasztjuk A egy tetszőleges sorát (vagy oszlopát),

ıennek minden elemét megszorozzuk az elemhez tartozó algebrai aldeterminánssal,
ımajd a kapott szorzatokat összeadjuk.

Az a_{ij} elemhez tartozó algebrai aldeterminány $(-1)^{i+j}A_{ij}$, ahol A_{ij} annakaz (n−1)-edrendű determinánsnak az értéke, amelyet A-ból az i. sor és j. oszlop kihúzásával kapunk

Vektorrendszer rangja

Definíció

Legyen $\mathcal{A}$ a vektortér egy vektorrendszere. Az $\mathcal{A}$ vektorrendszer rangja az általa generált altér dimenziója:

$$\operatorname{rang}(\mathcal{A}) = \dim(\mathcal{L}(\mathcal{A})).$$

Példa: $V = \mathbb{R}^3$, legyen $\mathcal{A} = \{u, v, w\}$, ahol

$$u = \begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix}, \quad v = \begin{pmatrix} 1 \\ 3 \\ 0 \end{pmatrix}, \quad w = \begin{pmatrix} 3 \\ 5 \\ 2 \end{pmatrix}.$$

Mivel $w = 2u + v$, ezért w benne van a másik 2 vektor által generált altérben. Viszont u és v lineárisan független, ezért $\operatorname{rang}(\mathcal{A}) = 2$.

Megjegyzés: Legyen V n-dimenziós vektortér, $\mathcal{A} = \{v_1, \ldots, v_m\} \subset V$. Ekkor $\operatorname{rang}(\mathcal{A}) \leq n$ és $\operatorname{rang}(\mathcal{A}) \leq m$.

Tétel

Egy vektorrendszer rangja nem változik, ha bármely eleméhez hozzáadjuk a többi elem tetszőleges lineáris kombinációját.

A V vektortér egy nemüres W részhalmazát V alterének nevezzük, ha W maga is vektortér, azaz zárt a vektor összeadásra és a skalárral való szorzásra.

Speciális mátrixok, inverz.

Definíció

Legyen A egy n-edrendű kvadratikus mátrix.

- A szimmetrikus, ha $A^T = A$,
- A ferdeszimmetrikus, ha $A^T = -A$.

Példák:

$$A = \begin{pmatrix} 2 & -3 & 4 \\ -3 & -1 & 7 \\ 4 & 7 & 0 \end{pmatrix} \qquad B = \begin{pmatrix} 0 & 2 & 1 \\ -2 & 0 & -5 \\ -1 & 5 & 0 \end{pmatrix}$$

Állítás – szimmetrikus és ferdeszimmetrikus mátrixok tulajdonságai

- Ferdeszimmetrikus mátrix főátlójában 0-k állnak.
- Szimmetrikus mátrixok összege szimmetrikus.
- Ferdeszimmetrikus mátrixok összege ferdeszimmetrikus.
- Szimmetrikus mátrixok szorzata nem feltétlenül szimmetrikus. De ha A és B szimmetrikus és felcserélhetőek (azaz $AB = BA$), akkor AB is szimmetrikus.

Mátrixok inverze

Definíció

Azt mondjuk az A n-edrendű négyzetes mátrixról, hogy invertálható, vagy létezik az inverze, ha létezik olyan B n-edrendű kvadratikus mátrix, hogy

$$AB = BA = E_n.$$

Tétel

Ha A invertálható, akkor az inverze egyértelmű. Jele: A^{-1}.

Példa:

$$A = \begin{pmatrix} 4 & 3 \\ 7 & 5 \end{pmatrix} \qquad A^{-1} = \begin{pmatrix} -5 & 3 \\ 7 & -4 \end{pmatrix}$$

Állítás – a mátrixinvertálás tulajdonságai

- Ha A invertálható, akkor A^{-1} is az és $(A^{-1})^{-1} = A$.
- Ha A és B invertálható és létezik AB, akkor ez is invertálható és $(AB)^{-1} = B^{-1}A^{-1}$.
- Ha A invertálható, akkor A^T is az és $(A^{-1})^T = (A^T)^{-1}$.

Mátrix, mint lineáris transzformáció.

Legyen V vektortér R felett. $\varphi: V \to V$ lineáris transzformáció, ha additív, azaz $\forall u,v \in V$: $\varphi(u+v) = \varphi(u) + \varphi(v)$; homogén, azaz $\forall v \in V$, $\lambda \in R$: $\varphi(\lambda v) = \lambda \varphi(v)$.

Lineáris transzformációk esetén nullvektor képe nullvektor.

Példák:Forgatások, tükrözések,λ-nyújtások.

Vetítések, pl.R_3 egy rögzített síkjára merőlegesen.

Identikus transzformáció: $\varphi(v) = v, \forall v \in V$.

Egy lineáris transzformációt egyértelműen meghatároz egy bázison való hatása, azaz ha $B = (b_1, b_2, ..., b_n)$ bázisa V-nek,$w_1, w_2, ..., w_n$ pedig tetszőleges vektorai a vektortérnek, akkor egyértelműen létezik olyan φ lineáris transzformáció, hogy $\varphi(b_i) = w_i$. Továbbá ha $v = \lambda_1 b_1 + \lambda_2 b_2 + \cdots + \lambda_n b_n$, akkor ennek φ általi hatása:$\varphi(v) = \lambda_1 w_1 + \lambda_2 w_2 + \cdots + \lambda_n w_n$.

Legyen V egy n-dimenziós valós vektortér,$B = (b_1, b_2, ..., b_n)$ bázisa V-nek, tekintsünk továbbá egy $\varphi: V \to V$ lineáris transzformációt. Ekkor φ-nek a B bázisra vonatkozó mátrixa az az n×n-es mátrix, amelynek i-edik oszlopában $\varphi(b_i)$-nek a B bázisra vonatkozó koordinátái állnak.
Példa: Legyen $\varphi: R_2 \to R_2$, $(x,y) \to \varphi(x,y) = (2x - y, -12x + 3y)$. φ mátrixa a természetes bázisban. $\varphi(e_1) = \varphi(1,0) = (2,-12)$, $\varphi(e_2) = \varphi(0,1) = (-1,3)$, ezért φ mátrixa ebben a bázisban

$$A_\varphi = \begin{pmatrix} 2 & -1 \\ -12 & 3 \end{pmatrix}$$

φ mátrixa ab$_1$= (1,1),b$_2$= (0,−1) bázisban. Ekkor$\varphi(b_1) = (1,-9)$és$\varphi(b_2) = (1,-3)$. Ezeket a vektorokat a (b_1, b_2) bázisban kell felírnunk:$\varphi(b_1) = (1,-9) = 1 \cdot b_1 + 10 \cdot b_2$, $\varphi(b_2) = (1,-3) = 1 \cdot b_1 + 4 \cdot b_2$.Ezért a mátrix:

$$[A_\varphi]_{(b_1,b_2)} = \begin{pmatrix} 1 & 1 \\ 10 & 4 \end{pmatrix}.$$

Lineáris transzformáció mátrixának alkalmazása

Állítás

Egy lineáris transzformáció különböző bázisokra vonatkozó mátrixainak megegyezik a rangja és a determinánsa.

Állítás

Ha φ mátrixa a $\mathcal{B}$ bázisban A, akkor φ hatása egy v vektoron: $\varphi(v) = Av$.

Példák: Forgatások és tükrözések mátrixa $\mathbb{R}^2$-ben a természetes bázisban:

$$\mathrm{rot}_\alpha = \begin{pmatrix} \cos\alpha & -\sin\alpha \\ \sin\alpha & \cos\alpha \end{pmatrix} \qquad \mathrm{refl}_\alpha = \begin{pmatrix} \cos(2\alpha) & \sin(2\alpha) \\ \sin(2\alpha) & -\cos(2\alpha) \end{pmatrix}$$

Így például a $v = \binom{2}{6}$ vektort az origó körül 60°-kal pozitív irányba elforgatva:

$$\begin{pmatrix} \cos 60^\circ & -\sin 60^\circ \\ \sin 60^\circ & \cos 60^\circ \end{pmatrix} \begin{pmatrix} 2 \\ 6 \end{pmatrix} = \begin{pmatrix} \frac{1}{2} & -\frac{\sqrt{3}}{2} \\ \frac{\sqrt{3}}{2} & \frac{1}{2} \end{pmatrix} \begin{pmatrix} 2 \\ 6 \end{pmatrix} = \begin{pmatrix} 1 - 3\sqrt{3} \\ \sqrt{3} + 3 \end{pmatrix}.$$

Sajátérték, sajátvektor.

Legyen φ:V→V lineáris transzformáció. Egy nem-nulla v∈V vektort φ sajátvektorának hívunk, ha ∃λ∈R: φ(v) =λv. Ekkor λ-t φ v-hez tartozó sajátértékének mondjuk.

Ha v sajátvektora φ-nek, akkor a hozzá tartozó sajátérték egyértelmű. Ha λ sajátérték, akkor a hozzá tartozó sajátvektorok halmaza altér: Lλ:={v∈V|φ(v) =λv} altér V-ben: a λ-hoz tartozó saját altér.

Egy φ lineáris transzformáció karakterisztikus polinomján a det(A−λEn) n-edfokú polinomot értjük, ahol n a tér dimenziója, A pedig φ mátrix a tetszőleges bázisban. Ennek gyökei éppen φ sajátértékei.

Példa sajátértékek, sajátvektorok meghatározására

Határozzuk meg az alábbi φ sajátértékeit és sajátvektorait!

$$\varphi\colon \mathbb{R}^2 \to \mathbb{R}^2,\ (x,y) \mapsto \varphi(x,y) = (2x - y, -12x + 3y)$$

Már láttuk, hogy φ mátrixa a természetes bázisban $\begin{pmatrix} 2 & -1 \\ -12 & 3 \end{pmatrix}$. Így φ karakterisztikus polinomja

$$\det(A - \lambda E_n) = \left| \begin{pmatrix} 2 & -1 \\ -12 & 3 \end{pmatrix} - \lambda \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} \right| = \left| \begin{pmatrix} 2-\lambda & -1 \\ -12 & 3-\lambda \end{pmatrix} \right| =$$
$$= (2-\lambda)(3-\lambda) - (-1)(-12) = \lambda^2 - 5\lambda - 6 = (\lambda+1)(\lambda-6).$$

Így a sajátértékek $\lambda_1 = -1$ és $\lambda_2 = 6$. Például a $\lambda_2 = 6$-hoz tartozó sajátvektorok:

$$\begin{cases} 2x - y = 6x \\ -12x + 3y = 6y \end{cases} \Rightarrow \begin{cases} -4x - y = 0 \\ -12x - 3y = 0 \end{cases} \Rightarrow \begin{pmatrix} x \\ y \end{pmatrix} = t \cdot \begin{pmatrix} 1 \\ -4 \end{pmatrix}, t \in \mathbb{R}.$$

A problémaredukciós reprezentáció és az ÉS/VAGY gráfok. Ismeretreprezentációs technikák, bizonytalanság-kezelés (fuzzy logika). A rezolúciós kalkulus. A logikai program és az SLD rezolúció. A logikai programozás alapvető módszerei.

A problémaredukciós reprezentáció és az ÉS/VAGY gráfok.

Gyakran előfordul, hogy egy problémát úgy próbálunk megoldani, hogy több külön-külön megoldandó részproblémára bontjuk. Ha a részproblémákat megoldjuk, az eredeti probléma megoldását is megkapjuk. A részproblémák megoldását további részek meg oldására vezetjük vissza, egészen addig, amíg csupa olyan problémához nem jutunk, amelyeket egyszerűségüknél fogva már könnyedén meg tudunk oldani. A probléma megoldásnak ezt a módját problémaredukciónak nevezzük. Redukció esetén a problémát addig bontom egyszerűbb problémákra, míg meg nem tudom őket oldani. Ezt redukciós operátorokkal valósítom meg. Redukciós operátorok értékkészlete a probléma hatványhalmaza.
A probléma problémaredukciós reprezentációja: <P, p, ε, R> (itt az ε csak az E írott változata, nem epsilon), ahol
- Az eredeti problémát, jelöljük most p-vel.
- Egy probléma részproblémákra bontása során a nyert részek az eredeti problémához hasonló, de annál egyszerűbb problémák. Jelöljük az így nyert problémahalmazt P-vel. Természetesen p∈P.
- P problémáinak összegyűjtése során törekszünk arra, hogy legyenek közöttük olyanok, melyeket meg tudunk oldani, vagy ismerjük a megoldásukat. Ezek a problémák az ún.egyszerű problémák. Az egyszerű problémák halmazát ε-vel jelöljük. ε ⊂P, hiszen p !∈ε, különben nincs megoldandó feladat.

Meg kell még adni a problémákat egyszerűsítő, illetve részekre bontó redukciós operátorokat. Egy redukciós operátor egy problémához azokat a (rész)problémákat rendeli hozzá, melyek egyenkénti megoldásával a probléma megoldása is előáll. Jelöljön a redukciós operátorok R véges halmazából r egy operátort. Ekkor
Dom(r) ={q|q∈P \ ε és r-alkalmazásának-előfeltétele(q)} és
Rng(r) ={r(q) ={q_1, . . . , q_m}|q∈Dom(r)és q_1, . . . , q_m∈P}.
Tehát egy redukciós operátor egy-egy problémához P egy-egy részhalmazát rendeli, így értékkészlete P hatványhalmazának valamely részhalmaza.

A definíció tehát: Legyen p egy probléma. Azt mondjuk, hogy a p problémát problémaredukciós reprezentációval írtuk le, ha megadtuk a 〈P, p,E,R〉 négyest, azaz
•a megoldandó p∈P problémát,
•a P !=∅ halmazt, a p problémához hasonló problémák halmazát,
•az egyszerű problémák ε ⊂P halmazát és
•a redukciós operátorok R !=∅ véges halmazát.
Jelölése: 〈P, p,E,R〉 .

Legyen a p probléma a 〈P, p,E,R〉 reprezentációval leírva és legyenek
Q={p_1, p_2, . . . , p_i, . . . , p_n}⊆P
Q′={p_1, p_2, . . . , p_{i-1}, q_1, q_2, . . . , q_m, p_{i+1}, . . . , p_n}⊆P
egy-egy problémahalmaz (n≥1, m≥1). (pi-ből lett q1...qm)

Azt mondjuk, hogy a Q problémahalmaz egy lépésben vagy közvetlenül redukálható a Q′ problémahalmazzá, ha van olyan r∈ R redukciós operátor,melyre $p_i \in Dom(r)$, és $r(p_i) = \{q_1, q_2, \ldots, q_m\}$. Ennek jelölése: $Q \lhd Q'$, illetve ha fontos, hogy az r redukciós operátor segítségével állítottuk elő Q-ból a Q′-t, akkor $Q \lhd r Q'$.

Legyen a p probléma reprezentációja 〈P, p,E,R〉 ,és Q, $Q' \subseteq P$. A Q-ból a Q′ redukálható, ha van olyan $P_1, \ldots, P_k \subseteq P (k \geq 2)$ véges problémahalmaz-sorozat, hogy $P_1=Q$, $P_k=Q'$ és $P_i \lhd P_{i+1}$ minden $1 \leq i \leq k-1$ esetén. Jelölése:$Q \lhd * Q'$.
Nyilvánvaló, hogy ha $P_i \lhd P_{i+1}$ minden $1 \leq i \leq k-1$ esetén, akkor van olyan $r_1, r_2, \ldots, r_{k-1}$ redukciós operátor-sorozat, hogy $P_i \lhd r_i P_{i+1} (1 \leq i \leq k-1)$. Ilyenkor azt mondjuk, hogy a Q problémahalmazt a Q′ problémahalmazzá az $r_1, r_2, \ldots, r_{k-1}$ redukciós operátorsorozat segítségével redukáltuk.
Jelölve:$Q \lhd * r1,...,rk-1 Q'$.

Legyen a p probléma problémaredukciós reprezentációja 〈P, p,E,R〉 . A p probléma megoldható ebben a reprezentációban, ha {p}csupa egyszerű problémából álló problémahalmazzá redukálható , azaz $\{p\} \lhd * r_1,...,r_l Q \subseteq E$. Ekkor az $r_1, \ldots, r_l$ redukciós operátorsorozatot tekinthetjük a probléma megoldásának.

(megj. könnyebb érthetőség kedvéért az állapottér reprezentációból ismert elérhetőséghez hasonlítsuk a redukálhatóságot)

A feladatunk lehet
•annak eldöntése, hogy megoldható-e a probléma az adott problémaredukciós reprezentációban,
•egy (esetleg az összes) megoldás előállítása,
•valamilyen minősítés alapján jó megoldás előállítása (a megoldások között különbséget tehetünk, pl. a megoldás költsége alapján).

Jóságot, vagy minőséget rendelhetek egy megoldáshoz annak költsége alapján (amely az egyszerű probléma megoldásának és az operátoroknak a költsége): redukálás és a redukció során előállt összes probléma megoldásának költsége + redukálás költsége.
Ha a megoldás párhuzamosítható és fontos a végrehajtási idő, akkor a párhuzamosítás miatt a leghosszabb végrehajtási időt kell figyelembe venni, hiszen azt kell megvárni.

Gráffal szemléltetve:

- csúcsok: a problémaosztály problémái
- start: megoldandó probléma
- célcsúcsok (levelek): egyszerű, megoldható problémák
- redukciós operátorok: problémát redukálnak probléma halmazzá.

p redukálásának eredménye n db különböző probléma, melyek egy ÉS élköteget alkotnak. Keresés során azt kell eldönteni, hogy melyik redukciót bontjuk tovább. Tehát az ÉS élkötegek VAGY kapcsolatban állnak egymással => ÉS/VAGY gráf.

Legyen a p probléma a 〈P, p,E,R〉 reprezentációval megadva. Ez a reprezentáció is egy irányított gráfot, ún. ÉS/VAGY gráfot határoz meg.
•A P problémahalmaz elemei (a problémák) a gráfcsúcsai. Vezessük be a q∈P probléma által definiált csúcsra az n_q jelölést. Ekkor a gráf csúcsainak halmaza $N=\{n_q | q \in P\}$.
•A gráf csúcsai közül kitüntetett szerepet játszanak a p problémát szemléltető ún.startcsúcs (jele:n_p vagy s)
•és az egyszerű problémákat szemléltető terminális csúcsok. A terminális csúcsok halmaza tehát:

$T=\{n_e|e\in E\}$.
•Egy $q\in P$ problémát szemléltető csúcsból irányított éleket húzunk az $q_1, \ldots, q_m\in P$ problémákat szemléltető $n_{q1}, \ldots, n_{qm}$ csúcsokba, amikor $\{q\}\triangleleft\{q_1, q_2, \ldots, q_m\}$. Ezek az élek összetartozónak tekinthetők: egy ÉS élköteget vagy hiperélt alkotnak. A gráf hiperéleinek halmaza tehát a következő: $E=\{(n_q,\{n_{q1}, n_{q2}, \ldots, n_{qm}\})|q, q_1, q_2, \ldots, q_m\in P$ és $\{q\}\triangleleft\{n_{q1}, n_{q2}, \ldots, n_{qm}\}\}$. Azt mondjuk, hogy az 〈N, s, T, E〉 irányított ÉS/VAGY gráf a p probléma 〈P,p,E,R〉 problémaredukciós reprezentációjához tartozó reprezentációs gráfja.

Legyen 〈N, s, T, E〉 a p probléma 〈P, p,E,R〉 problémaredukciós reprezentációjához tartozó reprezentációs gráfja. Pontosan akkor áll fenn a $\{q\}\triangleleft*\{q_1, q_2, \ldots, q_m\}$ reláció, ha a reprezentációs gráfban van az n_q csúcsából induló olyan hiperút, melynek levelei éppen az$\{n_{q1}, n_{q2}, \ldots, n_{qm}\}$csúcsok.

Tegyük fel, hogy $\{q\}\triangleleft*\{q_1, q_2, \ldots, q_m\}$. Azaz definíció alapján a reprezentációs gráfunkban egy $k-1$ hiperélből álló sorozatunk van, melyben az első hiperél a q-t szemléltető n_q csúcsból indul, minden következő hiperél kezdőcsúcsa valamely előző hiperél végcsúcsa, és minden csúcsból legfeljebb egy hiperél indul. Tehát a szemléltető részgráf egy hiperút. Továbbá a sorozat utolsó halmazának,P_k-nak a problémái azok, amiket nem bontottunk tovább, tehát az ezeket szemléltető csúcsok a hiperút levelei.
Most tegyük fel azt, hogy a reprezentációs gráf n_q csúcsából indul olyan hiperút, melynek levelei az$\{n_{q1}, n_{q2}, \ldots, n_{qm}\}$csúcsok. A sorozat minden $(n_i,\{n_{i1}, n_{i2}, \ldots, n_{imi}\})$ hiperéle egy redukciós operátoralkalmazást szemléltet: az n_i által szemléltetett problémát bontja a redukciós operátor az$\{n_{i1}, n_{i2}, \ldots, n_{imi}\}$csúcsok által szemléltetett problémákká. Tehát a hiperél sorozat egy redukciós operátorsorozat, mely első operátorát q-ra alkalmaztuk, az összes többit pedig, valamely megelőző operátor eredményeképpen előállt problémára. Legyen 〈N, s, T, E〉 a p probléma 〈P,p,E,R〉 problémaredukciós reprezentációjához tartozó reprezentációs gráfja. Pontosan akkor oldható meg p, ha van a reprezentációs gráfban a start-csúcsból induló olyan hiperút, melynek levelei terminális csúcsok. (Tehát egy hiperút, mely a start csúcsnál indul, és levelei csak megoldható, egyszerű problémákat szemléltetnek.)

Egy csúcsból vagy ÉS élek vagy VAGY élek indulnak. Az ÉS éleken belül az összes lépéssel foglalkozni kell, ezek a hiperélek.
A hiperél egy csúcs (a kezdőcsúcs) és egy csúcshalmaz (végcsúcsok, vagy befutó csúcsok) halmaza.
Hiperút: (látszatra egy gráf) a hiperélek vmilyen sorozata, ahol 2 feltétel teljesül:
- a kezdőcsúcsok mind különbözőek
- az elsőbeli élt kivéve minden hiperél kezdőcsúcsa megegyezik valamely őt megelőző hiperél végcsúcsával.

Címkézés során: m:megoldott
n: nem lehet a reprezentációban megoldani
f: folyamatban a megoldás keresése

Minden lépés után újracímkézünk az adatbázisban alulról felfelé. ÉS szülő akkor megoldható, ha minden gyermeke m címkéjű, VAGY szülő akkor megoldható, ha legalább 1 gyermeke megoldható, egyébként a szülő f és a startcsúcs is f. Akkor ér véget a keresés, ha a start csúcs m-re vagy n-re vált.

Ismeretreprezentációs technikák, bizonytalanság-kezelés (fuzzy logika).

Tudásreprezentációs technikák

Tudástípus	**Tudásábrázolás**
deklaratív	logikaszabály
struktúrált	szemantikus hálókeret
procedurális	stratégia,eljárás, függvény

I. Szabályalapú ismeretreprezentáció

Más néven **produkciós rendszerek**.
Első megfogalmazása a 40-es évekre tehető.
Simon és Newell alkalmazták először.
Mindmáig a leggyakrabban használt tudásreprezentációs módszer.

alkalmas a **köznapi gondolkodás** modellezésére és a szakértő tapasztalatait kifejező **heurisztikák** leírására

Szabályok segítségével fogalmazhatók meg a tárgyterület **tényei**, és a közöttük fennálló **relációk**, a meglévő tényekből új tényeket **következtető ismeretek**, a feladat megoldási stratégiájának módosítására szolgáló **metaismeretek**.

Egy szabály

HA feltétel AKKOR következmény

alakban egy ***ismeretelemet*** reprezentál, ahol a

feltétel: a szabály alkalmazásának feltételeit megadó (tény) állítás, vagy ilyenekből ÉS/VAGY kapcsolókkal képzett összetett kifejezés

következmény: a szabály alkalmazásának egy vagy több következményét írja le (akciók, műveletek, tevékenységek, érvényes állítások)

a munkamemória tartalmát módosító akciókat (adatelemek beírása, törlése, módosítása)

különböző eljáráshívásokat (amelyek a belső és a külső környezet közötti információcserét biztosítják)

a rendszer által vezérelt folyamatba történő beavatkozást (pl. egy kapcsoló bekapcsolását)

a felhasználótól való információkérést

HA feltétel
AKKOR következmény

Ha a **feltétel** teljesül, akkor a **következményben** előírt tevékenységek rendre végrehajtódnak.

A vezérlés

A vezérlő komponens **háromfázisú motor**ként működik:

1.Mintaillesztéssel megkeresi azokat a szabályokat, amelyek feltételrésze a munkamemória pillanatnyi tartalma fölött **igaz**, és ezeket a végrehajtásra alkalmas, ún. tüzelőképes szabályokat behelyezi egy konfliktus halmazba.
2.Kiválaszt a **konfliktushalmazból** egy szabályt végrehajtásra – a kiválasztásban a (beépített) vezérlési stratégiára támaszkodik).

3.Alkalmazza, végrehajtja a kiválasztott szabályt, vagyis **érvényesíti a szabály következményét.** Közben a vezérlő komponens figyeli a célt megfogalmazó *terminális feltétel bekövetkezését*, amikor is a végrehajtás **leáll**; ellenkező esetben **újra indul** az 1. fázis.

A 2. fázisban felhasznált **szabály-kiválasztási stratégia** sokféle lehet: előnyben részesülhetnek pl.
a munkamemóriába **legutóbb bekerült** (legfrissebben beírt) **adatra hivatkozó**, vagy
a **legbonyolultabb feltételrésszel** rendelkező szabályok,
történhet a választás a szabályokhoz rendelt **prioritás** szerint.

Mi a fuzzy logika?

A fuzzy logika leginkább a halmazalgebrai megközelítésmódban érthető meg. Gondoljunk vissza a hagyományos logika alapját képző halmazelméletre: legyen $\mathbf{A=\{a_1, a_2, a_3, \ldots, a_n\}}$ egy halmaz. A klasszikus logika szerint egy elem halmazba tartozása egyértelműen megállapítható, tetszőleges $\mathbf{a_k}$ elemről el tudtuk dönteni, hogy eleme-e az **A** halmaznak avagy sem! Tehát az hogy $\mathbf{a_k \in A}$ egyértelműen eldönthető. Ha beletartozik, úgy ezt egy logikai igaz, ha nem azt egy logikai hamis értékekkel jellemeztük. Egyszerűség kedvéért jelöljük a logikai igaz értéket 1-el a hamis értéket 0-val. Ekkor az, hogy egy elem beletartozik-e **A**-ba jellemezhető vagy egy 0-val, vagy egy 1-el.

A fuzzy logika abban hoz újat, hogy halmazba tartozás 0, illetve 1 értékei nem ennyire sarkallatosak, hanem köztes értékek is léteznek, amelyek megmutatják, hogy egy adott $\mathbf{a_k}$ elem mennyire tartozik bele a halmazba: nagyon, kissé, kevésbé, vagy egyáltalán nem. Így minden **A** halmazbeli $\mathbf{a_k}$ elemhez hozzárendelünk egy számot, általában 0 és 1 (néha -1 és 1 között), ami jellemzi az elem halmazba tartozásának mértékét. Tehát az **A** halmazunk fuzzyban az alábbi módon néz ki: $\mathbf{A=\{a_1^{(k1)}, a_2^{(k2)}, \ldots a_n^{(kn)}\}}$. A felső indexbe írt értékek a halmazelemekhez rendelt, halmazba tartozást jellemző számot jelöli. Vegyük észre, hogy ezek a számok a klasszikus halmazelméletben is jelen voltak, de értékük vagy 0 volt vagy 1, így külön nem is

tüntettük ezeket fel. Azt az elemet, amihez 0-át (vagy egy másik skálán -1-et) rendeltünk, fel sem soroltuk a halmaz elemei között.

Nézzünk erre egy példát. Legyen az **A** halmazunk az emberek cm-ben kifejezett testmagassága, és vegyük csak az egész értékeket.

A={130,131,132,…, 183, …, 250}

A klasszikus halmazelmélet szerint, ha meg akarunk határozni két részhalmazt, **M** jelölje a magas emberek halmazát, **L** az alacsony emberekét, akkor élesen kell találjunk egy elemet (például 170 cm), amelytől magasabb emberek az **M={170,171, … , 250}** halmazba tartoznak, míg az alacsonyabbak az **L={130,131, … 169}** halmazba. (A két részhalmaz **L**, **M** nem kell feltétlen diszjunkt legyen.)

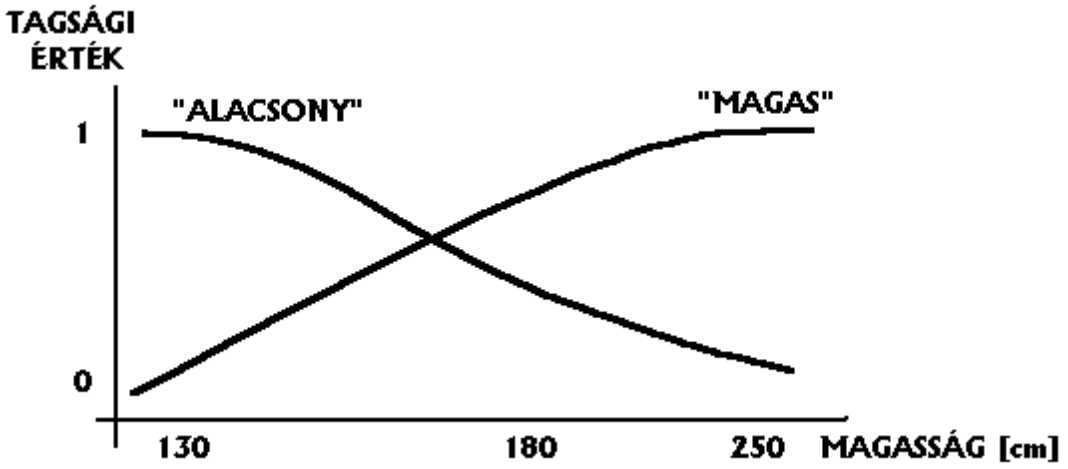
Az életben viszont ilyen éles határokat gyakran nem szabhatunk, azt mondjuk valakire, hogy “a körülbelül 155 cm magas illető, nagyjából alacsonynak mondható”. Tehát az állításban van egyfajta bizonytalansági tényező, “körülbelül”, illetve “nagyjából”. Az egyes elemehez rendelt értékek éppen ezt a bizonytalanságot hivatottak kezelni. Azt mondjuk, hogy az egyes részhalmazok elemeihez hozzárendelünk egy-egy számot. Például:

L={$130^{(1)}$, $140^{(1)}$, $150^{(1)}$, $160^{(0.8)}$, $170^{(0.5)}$, $180^{(0.1)}$, $190^{(0)}$}

M={$130^{(0)}$, $140^{(0)}$, $150^{(0)}$, $160^{(0.1)}$, $170^{(0.4)}$, $180^{(0.9)}$, $190^{(1)}$, $250^{(1)}$}

A két halmazban lehetnek teljesen különböző elemek is, és az elemekhez rendelt számok között, sem halmazon belül, sem két halmaz között semmilyen összefüggés nincs előírva, leszámítva azt, hogy “értelmes”, azaz szemantikai jelentéssel bíró adatoknak kell lenniük.

Ha mindezt a hozzárendelést függvényben jelenítem meg, tehát a halmaz elemeihez hozzárendelve ábrázoljuk a tagsági beletartozást jelölő számokat, akkor a tagsági függvényt ábráját kapjuk (1. ábra). A tagsági függvényeket folytonos összefüggéssel, de diszkrét értékekre (a halmaz elemeire) adják meg. A tagsági függvények alakjuk szerint lehetnek háromszög, harang, szigmoid, trapéz, egyoldalú trapéz, fűrészfog, stb… alakúak.



1. ábra: Tagsági függvények

2. Fuzzy műveletek

Ahogy a klasszikus halmazt kiegészítettük tagságot jelölő számokkal, úgy természetesen újra kell értelmeznünk a klasszikus halmazműveleteket (unió, metszet, negáció, vagy ha fuzzy halmazok tényeket szimbolizálnak AND, OR, NOT műveleteket).

Két fuzzy halmaz AND (metszet) művelete az a halmaz, amely a két argumentum halmaz közös elemeit tartalmazza, minden elemet véve a legkisebb előforduló beletartozási értéken.

M AND L={$130^{(0)}$, $140^{(0)}$, $150^{(0)}$, $160^{(0.1)}$, $170^{(0.4)}$, $180^{(0.1)}$, $190^{(0)}$}={$160^{(0.1)}$, $170^{(0.4)}$, $180^{(0.1)}$}

Két fuzzy halmaz OR (unió) művelete az a halmaz, ami minden előforduló elemet tartalmaz a lehető legnagyobb beletartozási értéken véve.

M OR L={$130^{(1)}$, $140^{(1)}$, $150^{(1)}$, $160^{(0.8)}$, $170^{(0.5)}$, $180^{(0.9)}$, $190^{(1)}$, $250^{(1)}$}

Egy halmaz negáltján azt a halmazt értjük, mely tartalmazza az összes elemet, de az eredmény halmaz elemeinek tagsági értékeit kivonjuk 1-ből.

A rezolúciós kalkulus.

Az ítéletlogika eldöntésproblémájának két megfogalmazása:

a. $A1 \supset A2 \supset ... \supset An \supset B$ formula tautológia-e, illetve

b. $\{A1, A2, ..., An, \neg B\}$ formulahalmaz kielégíthetetlen-e.

Ha a b pontban szereplő halmaz formuláit konjunktív normálformára hozzuk, akkor már egy klózhalmaz kielégíthetetlenségének kérdéséről beszélünk. Egy nulladrendű klózhalmaz kielégíthetetlen, ha bármely interpretáció mellett van a halmaz elemei között legalább egy hamis igazságértékű klóz. Egy klóz pontosan akkor hamis igazságértékű egy interpretációban, ha minden literálja is hamis. Tehát akkor és csakis akkor lesz hamis

igazségértékű a klóz, ha a klóz negált ítéletváltozói igaz, a negálatlanok pedig hamis igazságértéket vesznek fel.

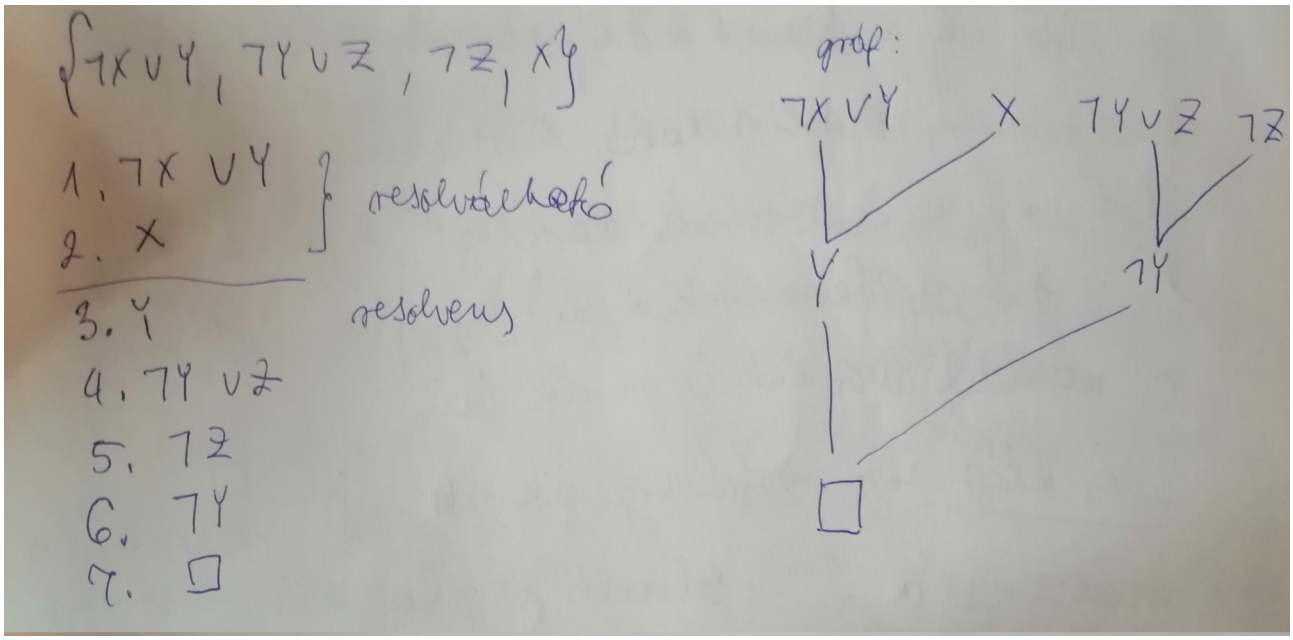
Legyen S egy véges klózhalmaz, és rögzítsük ítéletváltozóinak egy bázisát. Építsük fel a bázishoz a szemantikus fát. S egy C klózának *illesztése* a szemantikus fára azt jelenti, hogy megkeressük az(oka)t az ága(ka)t, amelyekben minden C-beli literálnak éppen a *komplemense* valamely él címkéje (egy literál komplemense a literállal azonos alapú, de ellentétesen negált literál, azaz X komplemense !X és !X komplemense X). Az ezeken az ágakon megadott interpretációkban a C klóz hamis igazságértékű. Illesszük most az S klózhalmaz minden klózát a szemantikus fa ágaira. Ekkor azt mondjuk, hogy elkészítettük az S *klózhalmaz szemantikus fáját*.

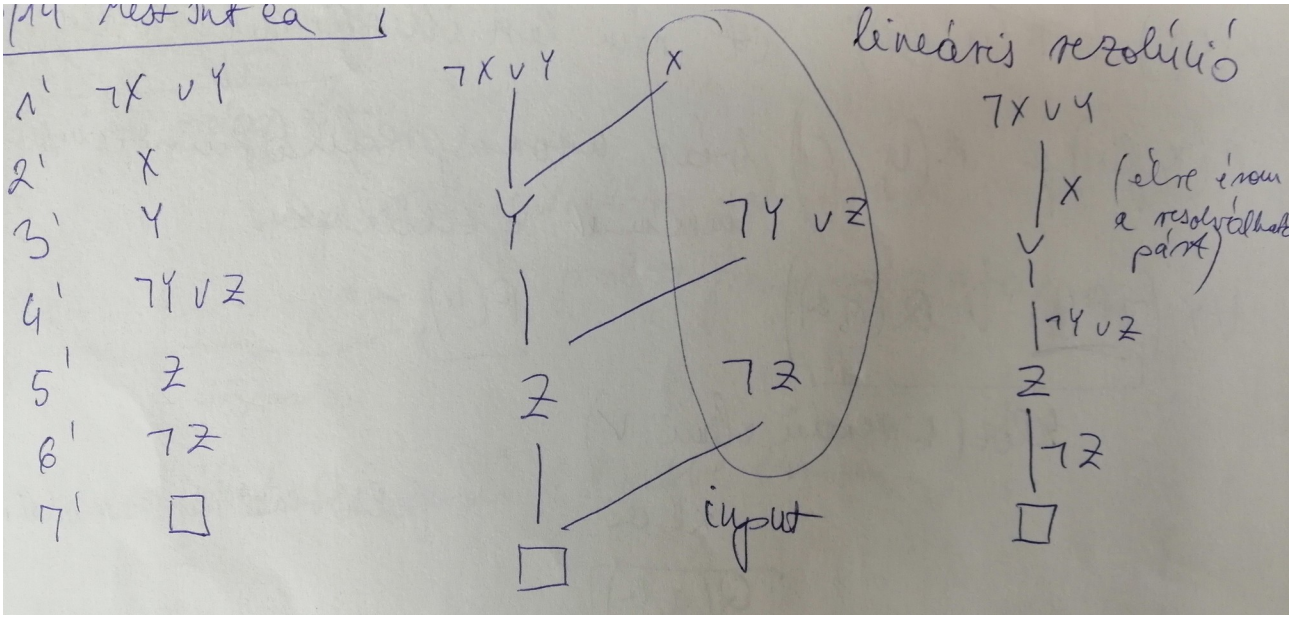
Speciális alakú formulákkal, klózokkal dolgozik, melyek mind elemi diszjunkciók.
Literáljai az atomi és a negált formulák.
Literálok diszjunkciója a klóz.
Komplemens literálpár a + és – literálok.

Klózokat úgy találunk, ha konjunktív normálformára hozunk. Klózok diszjunkcióláncára alkalmazható a rezolúciós kalkulus.

Rezolúciós levezetési szabály: 2 klóz, mely pontosan 1 komplemens literálpárt tartalmaz, rezolválható. A rezolvens 1 újabb klóz, mely a 2 resolválhatóból keletkezik úgy, hogy a komplemenseket kihagyjuk. (logika példatár 12.I.5.definíció, ahol C_1' maradék diszjunkció)
Az üres klóz nem tartalmaz literált, ez egy kielégíthetetlen formulát szimbolizál.
Resolvens a 2 resolvált logikai következménye, tehát C_1 és C_2 logikai következménye a C_1'vC_2'.

<u>Levezetés</u>
Szükséges 2 resolválható klóz.
Definíció: olyan klózsorozat, melyben minden klóz S-beli (kezdeti), vagy az előzőek resolvense. 1 klóz önmagának a következménye. Tehát egy levezetett klózhalmaz következménye az input halmaznak.
Ha az S kielégíthetetlen, akkor az üres klóznak van belőle levezetése. Ekkor teljes a rezolúciós kalkulus.



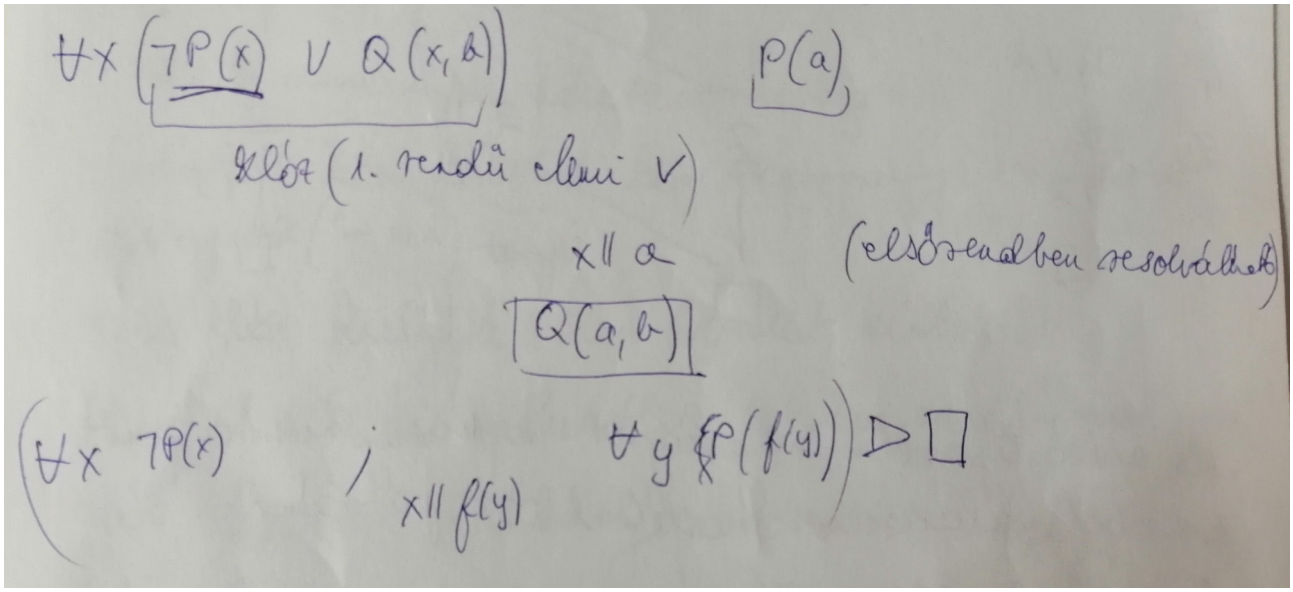


Ha az utolsó lépésben megkapott resolvenshez az input halmazból választunk mellékklózt párként, akkor lineáris input rezolúciót kapunk. Ez nem teljes rezolúciós stratégia, de jól irányít. Speciális klózok, Horn klózok (maximum 1 negálatlan literált tartalmaznak) esetén teljes.

Az olyan klózokat, amelyek legfeljebb egy nemnegált literált tartalmaznak, *Horn-klózoknak* nevezzük. A *Horn-formulák* pedig azok a formulák, melyek konjunktív normálformája Horn-klózok konjunkciója. Bebizonyították, hogy a lineáris inputrezolúciós stratégia Horn-formulák esetére (szokás azt is mondani, hogy a *Horn-logikában*) teljes.

Elsőrendű klózok: Univerzálisan kvantált, prenex alakú formulák. Minden változó szerint kvantáltak (zártak). Kvantormentes rész: elemi diszjunkció.
Alkalmazhatok változóhelyettesítést, hogy ugyanazt az atomot kapjam. Termhelyettesítést hasonlóképpen alkalmazhatok.

PL:
$R(c,x)\Theta = R(y, f(z))\Theta$
$R(c, f(z)) = R(c, f(z))\Theta \iff$ $y||c$
$x||f(z)$ illesztő helyettesítés



Tény: Horn klóz ilyen alakkal: $\forall x_1 ... \forall x_k B$

$\forall(!(A_1 \& ... \& A_n) || B)$
$(A_1 \& ... \& A_n) \rightarrow B)$

Legfeljebb 1 pozitív literál azt is jelentheti, hogy minden formula negált a Horn klózban.
Pl. $H_1 \& ... \& H_m \rightarrow C$ is Horn klóz.

Ha egy Horn formulához hozzátesszük a negáltját, és ez kielégíthetetlen, akkor tényt igazolunk. Ez a prolog programozás alapja. Resolválható párt tények és szabályok közül választunk.

A logikai program és az SLD rezolúció. A logikai programozás alapvető módszerei.

A logikai program egy tudást ír le, melyet a gép a neki feltett kérdésekre való válaszoláshoz felhasznál. A logikai program állítások gyűjteménye, melyek alkalmazási sorrendjéről a kérdés alapján a futtatókörnyezet dönt. (Prolog esetén az állítás tény vagy szabály.)
állítás:a logikai program része,
levezetési szabály: ezt alkalmazva a futtatókörnyezet újabb állításokhoz jut a már meglevőekből,
implikáció: művelet az elsőrendű logikában

Tiszta Prolog-programok (pure Prolog): a meta-logikai eljárások (pl. var/1, findall/3, setof/3, freeze/2) használatát, a vágót, a feltételes szerkezetet (A-> B ; C) és a negálást megtiltjuk. Nem szabad továbbá végtelen struktúrákat létrehozni (pl. X = f(X,Y)). Az egyszerűség kedvéért a diszjunkciót (';'/2) mellőzzük: ez az eljárások szétdarabolásával könnyen kiváltható. Ha egy ilyen Prolog-program nem tartalmaz se végtelen választási pontot, se végtelen ciklust (minden hívás véges időben lefut), akkor egy Prolog klózon belül szabadon cserélgethetjük a célok sorrendjét, ez nem befolyásolja a kérdésre adott választ. Ekkor egy „P :- Q, R, S." alakú Prolog klóz egy implikációnak felel meg: ha Q,R és S igaz, akkor P is. Formálisan:$P \leftarrow (Q \wedge R \wedge S)$.
Egy állítást akkor nevezünk igaznak, ha eljuthatunk hozzá az ismert dolgokból helyes logikai lépések egymásutánjával. A gép akkor döntötte el egy B állításról, hogy igaz-e, ha vagy B-hez, vagy az ellentétéhez (¬B-hez) így eljutott (és a köztes lépéseket kérésre be is tudja mutatni).

A Prolog működése felfogható úgy, hogy a B állítás (kérdés,query) negáltjáról próbálja meg bebizonyítani, hogy hamis, és a negáltra adott ellenpélda (változó-behelyettesítések) a B igaz voltát bizonyítja.
P logikai programban pontosan meghatároztuk a számára az ismert dolgokat, továbbá a bizonyítandó vagy cáfolandó B állítást is megadtuk neki. Általában a P programot a programozó, a B állítást pedig a felhasználó fogalmazza meg.
Nyelve: elsőrendű logika.
Ún. term építhető konstansokból, változókból és függvény-alkalmazásból, például apja (felesége(ernő, X)) egy term (jelentése lehet: Ernő X-edik házasságában szerzett apósa). Egyatomi formula(más néven literál) egyetlen predikátumból áll (az argumentumok termek), az egyéb formulák pedig atomiakból építhetők fel változókból,konstans-jelekből, függvényjelekből és predikátumjelekből. A logikai program (P) formulák egy véges halmaza.
Nem lehet leírni vélekedéseket, időbeli kijelentéseket), magasabb rendű (értsd: függvényekre vonatkozó) állításokat és – szerencsétlen predikátum választás mellett – lezártakra vonatkozó állításokat.
A gép nem ismeri az interpretációt (ez csak a programozó és a felhasználó fejében létezik), ezért minden következtetés, amit a gép – formálisan – levon, érvényes bármely interpretációban, tehát a felhasználó fejében levő, ún. szándékolt interpretációban is.
P logikai program állításai zárt formulák (nincs bennük szabad változó), melyek a világra vonatkozó tudást tartalmazzák.
A bizonyítandó B állítás is zárt formula P-ben. A gép feladata megtalálni a bizonyítást, ami a P-ben található formulákból elemi logikai lépésekkel eljut B-ig. Ezt általában indirekt bizonyítással végzi, vagyis felteszi, hogy P összes formulája és¬B is igaz, és ebből ellentmondásra jut. Az ellentmondás egy olyan (egyszerű) zárt formula, ami minden interpretációban hamis. Ilyen például bármely $F \wedge \neg F$ alakú formula, ahol F formula.
Rögzítve van néhány levezetési szabály, és ezeket próbálja alkalmazni a már ismert, igaz, zárt formulák némelyikére. A szabály egy új formulát eredményez. Direkt bizonyítás esetén ha az új formula megegyezik B-vel, akkor kész a bizonyítás, ellenkező esetben felveszi a már ismert formulák közé, és folytatja egy újabb levezetési szabály-alkalmazással. Az SLD-rezolúcióban is használt indirekt bizonyításban a leállási feltétel az azonosan hamis formulához való érkezés.

Híres levezetési szabály a modus ponens: ha F és $F \rightarrow G$ már ismert, igaz, zárt formulák, akkor G is igaz. Híres még a példányosítás: ha$\forall XF(X)$ ismert, igaz, zárt formula és t egy term, ami nem tartalmazza X-et, akkor F(t) is igaz. Híres még a rezolúció alaplépése: ha $F \vee G$ és $\neg F \vee H$ ismert, igaz, zárt formulák, akkor $G \vee H$ is igaz. Számtalan levezetési szabály felsorolható még, és mindegyikről könnyű ellenőrizni, hogy helyes, vagyis alkalmazásával igaz formulákból igaz formulákat kapunk, bármilyen modellben.
A rezolúció alaplépése önmagában nem teljes, de ha előtte átalakítjuk a formulákat (velük ekvivalens más formulákká, szükség esetén a nyelvet is bővítve), és megengedünk változó-helyettesítéseket is,akkor már teljes.
Ha tehát a levezetési szabályaink helyesek és teljesek, akkor minden igaz (vagyis P-ből következő) állítás bebizonyítható velük, és egyetlen hamis (vagyis az ellentéte P-ből következő) állítás sem bizonyítható be.
Bármely P (véges) logikai program, amelynek nyelve tartalmazza a természetes számokat (vagy bármely ennél bonyolultabb struktúrát) ellentmondásos vagy található egy benne eldönthetetlen állítás. Röviden: szinte bárhogy is írunk logikai programot, mindig lesz hozzá eldönthetetlen állítás.

Definit klózok
P logikai programban csak $\forall . . . A_0 \leftarrow (A_1 \wedge A_2 \wedge \cdots \wedge A_n)$ alakú formulákat engedünk meg, ahol minden A_i atomi formula (n= 0 is megengedett, ekkor A_0 feltétel nélkül igaz), a bizonyítandó B állítás pedig $\forall . . . B_0 \wedge B_1 \wedge \cdots \wedge B_m$ alakú kell legyen. A bizonyítás módszere a később ismertetett SLD-rezolúció lesz, ami „Linear resolution for Definite clauses with Selection function”-t jelent. Itt

a definit klóz(definite clause), röviden klóz fogalom a P program formuláinak fent bemutatott szintaxisát jelöli, A_0 a klóz feje, a többi formula pedig a klóz törzse. A lineáris kifejezés arra utal, hogy a B állításból származó klóz minden rezolúciós lépésben részt vesz, a kiválasztási függvény pedig az a függvény lesz, ami a bizonyítás következő lépését, pontosabban a B′ atomi formuláját meghatározza (ettől a függvénytől függ, hogy a bizonyítás milyen gyorsan készül el).
A $\forall . . . A_0 \leftarrow (A_1 \wedge A_2 \wedge \cdots \wedge A_n)$ klóz ekvivalens a $\forall . . . A_0 \vee \neg A_1 \vee \neg A_2 \vee \cdots \vee \neg A_n$ formulával, vagyis egy olyan diszjunkcióval, amiben 1 kivétellel az összes atomi formula negálva van. Indirekt bizonyítással a P-beli klózokon túl feltesszük, hogy a B formula hamis, tehát $\forall . . . \neg B_1 \vee \neg B_2 \vee \cdots \vee \neg B_m$) igaz. Ez tehát egy olyan diszjunkció, ahol az összes atomi formula negálva van. M= 0 esetén ezt a formulát azonosan hamisnak tekintjük.
Az SLD-rezolúció nevű bizonyításkereső módszer határozott klózokból gyárt újabb határozott klózokat. Az általános rezolúció diszjunktív klózokkal teszi ugyanezt. Egy diszjunktív klózban tetszőlegesen sok negálatlan atomi formula megengedett. Ez az alak bővebb a határozott klózénál például a $\forall . . . P \leftarrow (Q \wedge R \wedge S)$ logikai formulának a „P :- Q, R, S.” Prolog-klóz felel meg. A fordított irányú implikáció helytelen. Tehát például $\forall X \forall Y \forall Z$ nagyszülője(X, Z) $\leftarrow$ szülője(X, Y) $\wedge$ szülője(Y, Z) nem nyilatkozik arról, hogy ha a két „szülője” feltétel nem teljesül egyszerre (vagyis nem létezik megfelelő Y), akkor vajon X nagyszülője-e Z-nek vagy sem.

Herbrand-interpretáció
Ha a logikai program csak határozott klózokból áll, akkor biztosan nem tartalmaz ellentmondást, vagyis van modellje. A legszűkebb modell az a modell, amiben csak az igaz, ami a programból következik. Tehát például ha a járat/3 predikátum adja meg, hogy honnan hová mikor indul járat, akkor a modellben (jelen esetben a modell a valóságot jelenti) ne legyen más járat, mint ami a járat/3 forráskódjában fel van sorolva. Az alábbiakban ilyen modellt keresünk.

Herbrand-interpretációnak (H-ió) nevezünk egy olyan interpretációt, amiben különböző szintaktikájú tömör termek különbözőek. Pontosabban: H-ió a minden tömör termnek önmagát felelteti meg (tehát az ő világának elemei objektumai az adott nyelv tömör termei; apja(ernő) nem Ernő apját jelöli, hanem egyszerűen az apja(ernő) termet), a predikátumokra nincs kikötés. Tehát például 1−1·1 és 0 egy H-ió ban különböző, míg a szokásos interpretációban egyező. Ha egy H-ió modellje a programnak, akkor Herbrand-modellnek nevezzük. Mi a legszűkebb Herbrand-modellt (least Herbrand model) keressük, vagyis a predikátumokhoz szeretnénk hozzárendelni relációkat, hogy modellt kapjunk, és a relációknak a lehető legkevesebb elemük legyen. A „lehető legkevesebb” értelmezése: mivel Herbrand-modellek metszete (vagyis a bennük levő relációk metszete) is Herbrand-modell, ezért egy P(klózokból álló) program legszűkebb Herbrand-modellje legyen az összes Herbrand-modelljének metszete. P program logikai következményei (vagyis az összes belőle következő formulák) közül az atomi formulák közül a tömörek épp megegyeznek P legszűkebb Herbrand-modelljének relációival. a legszűkebb Herbrand-modell konstruálható fix-pontig menő iterációval. Legyen $\varnothing$ az az interpretáció, amelyben minden predikátum azonosan hamis, legyen ground(P) a P klózainak összes tömör példánya (tehát ahol azonos változó szerepel, azt azonos tömör kifejezéssel kell helyettesíteni). $T_P(I)$ ekkor egy függvény, ami az I interpretáció relációit bővíti: $T_P(I) := \{A_0 | (A_0 \leftarrow (A_1 \wedge A_2 \wedge . . . \wedge A_n)) \in \text{ground(P)}$ és minden $A_i \in I\}$. A T_P függvényt iterálva egyre bővebb interpretációt kapunk. Ha $T_P(I') = I'$, akkor I′ a legszűkebb Herbrand-modellje P-nek.

Egyesítés
A gép a bizonyítás keresése során a rezolúció alaplépését ismételgeti. Ez a P program klózait bővíti egy olyan új klózzal, melyet a már ismert klózokból az egyik levezetési szabályt egyszer alkalmazva meg lehet kapni. A rezolúció alaplépése vázlatosan a következő(ismétlés): ha van egy $A \vee P$ és egy $\neg B \vee Q$ alakú klóz, ahol A és B atomi formulák, P és Q pedig diszjunktív klózok (0 vagy több atomi formulával), akkor az új klóz $P' \vee Q'$ lesz.

SLD-rezolúció esetén az A∨P mindig az eredeti programból jön, ahol A egy klóz feje (A=A0). Arról, hogy melyik klózfej legyen a sok illeszkedő közül, a bizonyítási eljárás specifikációja dönt (ez nem a kiválasztási függvény!). /B∨Q pedig mindig bizonyítandó állítás negáltjából származó formula – hogy ebből melyik atomi formula lesz B, arról a kiválasztási függvény dönt.
Az A és B atomi formuláknak egyesíthetőknek kell lenniük. Ez azt jelenti, hogy létezik egy olyan változó-helyettesítés (substitution), amely A és B változóihoz termeket rendel, és a helyettesítés után kapott A′ és B′ megegyezik. Ha több alkalmas egyesítő (helyettesítés) is van, akkor azok közül a legáltalánosabbat (mgu, most general unifier) választjuk. Egy f helyettesítés általánosabb h-nál,ha létezik g helyettesítés, hogy f és g egymás után elvégzése épp h (vagyis f∘g=h). Tehát először az A∨P formulán elvégezve a helyettesítést kapjuk A′∨P′-t, majd /B∨Q-n elvégezve kapjuk /A′∨Q′-t, és ezeken végrehajtva egy rezolúciós lépést kapjuk P′∨Q′-t.
Ha a rezolúció sikerrel zárul, vagyis találtunk egy ellenpéldát, akkor kövessük végig, hogy milyen helyettesítéseket alkalmaztunk az ellentmondáshoz való eljutás során. Ezeket az eredeti, bizonyítandó B állításra alkalmazva megkapjuk a legáltalánosabb ellenpéldát. (Ez az ellenpélda megegyezik azzal, amit a Prolog kiír siker esetén.)

A fenti program mellett az `utazhat(X).` hívás kétszeresen is sikerül, mindkétszer X=c behelyettesítést adva.

```
utazhat(X) :- van_jegye(X).
utazhat(X) :- nyugdíjas(X).
utazhat(X) :- vak(Y), kutyája(Y,X).
vak(a).
vak(b).
kutyája(a,c).
kutyája(b,c).
```

A rezolúciós bizonyítás az alábbi klózokból indul:

utazhat(X)∨¬van_jegye(X) (1)
utazhat(X)∨¬van_jegye(X) (1)
utazhat(X)∨¬nyugdíjas(X) (2)
utazhat(Y)∨¬vak(X)∨¬kutyája(X, Y): vakvezető kutya (3)
vak(a) (4)
vak(b) (5)
kutyája(a, c) (6)
kutyája(b, c) (7)
¬utazhat(X) (8)

Az új klózok:

¬vak(X_1)∨¬kutyája(X_1, X_2): (3) és (8),$X_2=Y_1$ (9)
¬kutyája(a, X_2): (9) és (4),X_1=a (10)
□: (10) és (6),X_2=c (11)
Itt már készen van a bizonyítás, de a Prolog további ellenpéldákat keres:
¬kutyája(b, X_2); (9) és (4):X_1=b (12)
□: (12) és (6):X_2=c (13)

Az első ellentmondásban az X=X_2=c, és a másodikban is az X=X_2=c ellenpéldát kaptuk.

Két kifejezés pontosan akkor egyesíthető, ha külön-külön tömörré tehetők úgy, hogy ugyanazt a tömör termet kapjuk. Másképpen: két term pontosan akkor egyesíthető, ha változólekötések után Herbrand-interpretációban egyenlővé tehetők. Az egyesítés e módja a lehető legóvatosabb:van olyan interpretáció (és modell), ahol 1−X és 0 egyenlővé tehetők, de mivel olyan interpretáció is van, ahol nem, ezért inkább nem egyesítjük a rezolúcióban. (Emlékeztető: a rezolúció csak olyan állítást tud bebizonyítani, ami minden modellben igaz.)

A Prolog-os egyesítő algoritmus megengedi az X=g(X) és T=f(T, Y) kapcsán előkerülő, végtelen termet eredményező egyesítéseket. A végtelen term a matematikában nem szabályos, és a Prolog csak azért engedi meg, mert a unify_with_occurs_check/2 túl lassú (könnyű n db egyenlőséget felírni, amelyben az előfordulás-ellenőrző egyesítés 2^n idejű).
Előfordulás-ellenőrzés nélkül az SLD-rezolúció nem helyes! Például a T-vel és f(T, Y)-nal egyszerre egyesíthető termeket bizarrnak nevezzük, és egy Y term őrületes, ha van olyan bizarr term, aminek ő a második argumentuma, akkor nyilvánvaló, hogy nem létezik őrületes term (mivel ekkor létezne bizarr term is, tehát létezne végtelen term, ami a matematikában nem igaz). Az alábbi Prolog-program

```
effes(f(T,_Y),T).
bizarr(T) :- effes(T,T).
őrületes(Y) :- bizarr(f(_X,Y)).
```

mellett az őrületes(Y). cél előfordulás-ellenőrzés mellett helyesen meghiúsul, míg nélküle sikeres lesz, egyesítés nélkül, vagyis azt kaptuk, hogy minden term őrületes.

SLD Rezolúció
Tekintsük a P program és a ¬B állítás ∨-okkal felírt alakját. Ha a gép el tudja érni, hogy a B-beli diszjunkcióból fogyjanak el az atomi formulák, tehát váljon üressé, akkor eljutott egy hamis formuláig (□), tehát az indirekt bizonyítás sikeresen befejeződött. Ehhez a gép B-ben cserélgetni kezdi a formulákat: kiválaszt egy$\neg B_j$ formulát, és kicseréli egy vagy több negált atomi formula diszjunkciójára. (A régi B-ről elfeledkezik, már soha többet nem fogja használni.) Ezáltal B szerkezete megmarad, csak a benne található atomi formulák változnak. A cserélgetés során B hossza nőhet is, de a végcél az, hogy az összes formula elfogyjon belőle.
Az SLD-rezolúció alaplépése a következő (ismétlés): kiválaszt egy $\neg B_j$ formulát a legutóbbi csupa negatív atomi formulát tartalmazó formulából, és kiválaszt egy A_0 klózfejet a P programból. Ha B_j és A_0 megegyezik (vagy egyesíthetőek, lásd később), akkor $\neg B_j$-t kicseréli a klóz törzsére (csupa $\neg A_i$ diszjunkciója). Vegyük észre,hogy ez a lépés a rezolúció alaplépése. Az egyesítés előtt a program klózban átnevezi a változókat, hogy ne ütközzenek a B-beli változókkal. Továbbá a klózok egyesítése változást okoz a formulákban szereplő változókban: ezt a változást véghez kell vinni a keletkező B összes atomi formuláján, tehát valójában nem csak B_j változik, hanem a többi atomi formula is.

Mi történik akkor, ha a bizonyítandó B állítás nem következik a programból? Ezt úgy érzékeli a gép, hogy az SLD-rezolúció alaplépését már sehogy sem tudja véghez vinni, vagyis talál egy olyan B_j-t, ami semelyik klózfejre nem illeszthető. Ekkor visszatér egy korábbi, az előző lépés(eke)t megelőző B-re, és az ott próbálja másképpen alkalmazni az alaplépést. Tehát egy visszalépéses keresés (backtrack) jön létre. Ha végül minden ágon elakad, akkor megáll, és kiírja, hogy az állítás nem következik programból. Elképzelhetők azonban végtelen ágak is, ekkor az SLD-rezolúció nem áll le, a gép végtelen ciklusba kerül.
Az alaplépés alkalmazásakor B_j és A_0 szabadon választható (feltéve, hogy egyesíthetőek). A gép azt fogja választani, amit a kiválasztási függvény előír. Ily módon a bizonyítás megtalálásának hatékonysága (és végessége) a kiválasztási függvénytől is függ. A Prolog is SLD-rezolúciót futtat, kiválasztási függvénye mindig B_1-et választja, és az azonos eljáráshoz tartozó klózokat a visszalépés során a programban előfordulásuk sorrendjében próbálja végig. Ez egy elég buta megoldás (és néha fölöslegesen vezet végtelen ciklushoz), de a memóriahasználat és egyéb implementációs szempontok miatt ezt választották.
Tekintsük az alábbi Prolog-programot:

```
nagyapja(X,Z) :- apja(X,Y), szülője(X,Y).
szülője(X,Y) :- apja(X,Y).
szülője(X,Z) :- apja(X,Z).
apja(a,b).
```

```
anyja(b,c).
```

Az SLD-rezolúció menetét, a gép időbeli működését egy levezetési fában (SLD-fa) ábrázolhatjuk. A fa gyökere a B állítás (diszjunktív alakja), a további csúcsok pedig a B állítás módosításai. Egy B′ csúcsból él megy a B″ csúcsba (és az él címkéje a kiválasztott atomi formula sorszáma és a kiválasztott klóz sorszáma), ha B′-ből B″ egyetlen alaplépéssel elérhető. A fa levelei □ címkéjűek (ekkor az indirekt bizonyítás eljutott az ellentmondásig), vagy egyéb diszjunkciók (még nem teljesen megvizsgálva vagy kiderült, hogy alaplépés innen nem lehetséges). A bizonyítás maga egyetlen gyökértől levélig menő út a fában. A fa többi ága ekkor nem érdekes. Ha az összes megoldásra szükség van, akkor az összes ilyen utat és a rajtuk végrehajtott változó-behelyettesítéseket kell tekinteni. A fenti példaprogram esetén a `nagyapja(a,X).` kérdés levezetési fája(póriasan):

```
nagyapja(a,X).
     apja(a,Y0), szülője(Y0,X).
          szülője(b,X).
               apja(b,X).
                    nyelő, nem lehet továbblépni
               anyja(b,X).
                    (üres csúcs: hamis)
```

Egyetlen bizonyítást (gyökér→□ út) ábrázolhatunk bizonyítási fában is. Üres gyökeréből él megy a B állítás atomi formuláiba. Belső csúcsaiban két, egymással egyesíthető atomi formula szerepel. A levelekben egyetlen atomi formula van, ez egyesíthető egy tényállítással. Egy csúcsból kifelé induló élek az eredeti B_j+A_0 csúcsból az A_1+?...A_n+? csúcsokba mennek. A bizonyítás akkor teljes, ha elérkeztünk egy levélben egy tényállításig. A bizonyítási fa konzisztens, ha a csúcsokban levő egyesítések egyszerre elvégezhetők. A fenti példaprogram esetén a `nagyapja(X,Z).` kérdés levezetési fája (póriasan):

```
nagyapja(X0,Z0)
     apja(X0,Y0) = apja(a,b)
          igaz
     szülője(Y0, Z0) = szülője(X1, Y1)
          anyja(X1, Y1) = anyja(b,c)
               igaz
```

Képzeljünk el egy végtelen levezetési fát, melyben minden lehetséges csúcs és él szerepel. Az SLD-rezolúció ekkor a fában egy útkeresés a gyökértől valamely csúcsig. Ha létezik csúcs, akkor ahhoz szélességi kereséssel véges időben el lehet jutni. (Ha nem létezik záró csúcs, akkor lehet, hogy végtelen ideig tart a keresése, például a `p :- p.` program esetén egy végtelen hosszú út a fa.) A Prolog mégsem szélességi, hanem mélységi keresést használ, aminek hátránya, hogy végtelen ciklusba eshet. Ennek az az oka, hogy a szélességi keresés memóriaigénye nagy (az előző szint összes B′ állapotát a memóriában kell tartani), továbbá egy előrelépés lassú.

Miért szűkebb az SLD-rezolúció az általános rezolúciónál? Azért, mert az általános rezolúció diszjunktív klózaiban (és a B állításban is) tetszőleges számú negált és nem negált atomi formula szerepelhet. Az általános rezolúció kiválaszt két tetszőleges klózt (tehát az egyik nem mindig a B′), és belőlük egyesíti egy A és egy /A atomi formula argumentumait, majd végrehajtja a rezolúció alaplépését, egy új formulát hozva létre. Tehát A∨P és ¬A∨Q-ból P′∨Q′ keletkezik (a vessző az egyesítés hatását mutatja). Az általános rezolúciónak tehát több választási lehetősége van, ezért lassabban jut el a megoldáshoz (□).

Bővebben:

https://arato.inf.unideb.hu/aradi.bernadett/files/dimat/DiMat%20-%20prezi.pdf Lineáris algebra

https://users.iit.uni-miskolc.hu/~radai/MI/fuzzy.htm

https://arato.inf.unideb.hu/kovacs.zita/tr_lev.html tr_ea_2

https://gyires.inf.unideb.hu/KMITT/a02/ch06s03.html rezolúciós kalkulus

https://gyires.inf.unideb.hu/KMITT/a02/ch07s02.html logikai programozás és SLD rezolúció

https://www.cs.bme.hu/~szeredi/oktatas/vflp/vflp04/prologikai_talk.pdf logikai programozás és SLD rezolúció (rövidebb, én ezt használtam főként, ha valaki utánaolvasna)