



»Лекционен курс »Изкуствен интелект



Изводи в съждителната
логика»¹

Доказване на теореми в СЛ

- » Досега показвахме логически следствия посредством **проверка на модели**
 - > Изброяване на моделите и показване, че съжденията важат за всеки модел
- » Логическото следствие може да се реализира също посредством **доказване на теореми**
 - > Директно върху съжденията в базата знание се прилагат правила за извод, с цел **конструиране на доказателство на желаното твърдение**
 - > Не се консултират моделите
 - + В определени случаи броят на моделите може да бъде много голям
- » Освен логическо следствие се нуждаем още от някои допълнителни концепции



Предимства

- » Ако броят на моделите е голям, тогава **доказването на теоремите може да бъде по-ефективно** от проверката на модела
- » Преди да се запознаем с детайлите на доказване на теореми, освен логическо следствие, се нуждаем още от следните допълнителни концепции:
 - > Логическа еквивалентност
 - > Валидност
 - > Удовлетвореност



Логическа еквивалентност

- » Две съждения α и β са **логически еквивалентни**, когато са верни в **едно и също множество на модели**
 - > Запис: $\alpha \equiv \beta$
- » Алтернативна дефиниция:
 - > $\alpha \equiv \beta$, тогава когато $\alpha \models \beta$ и $\beta \models \alpha$
- » В логиката същата роля като аритметичната идентичност в класическата математика
- » Пример:
 - > $P \wedge Q$ и $Q \wedge P$ са логически еквивалентни



Логически еквивалентности

$$(\alpha \wedge \beta) \equiv (\beta \wedge \alpha) \quad \text{commutativity of } \wedge$$

$$(\alpha \vee \beta) \equiv (\beta \vee \alpha) \quad \text{commutativity of } \vee$$

$$((\alpha \wedge \beta) \wedge \gamma) \equiv (\alpha \wedge (\beta \wedge \gamma)) \quad \text{associativity of } \wedge$$

$$((\alpha \vee \beta) \vee \gamma) \equiv (\alpha \vee (\beta \vee \gamma)) \quad \text{associativity of } \vee$$

$$\neg(\neg\alpha) \equiv \alpha \quad \text{double-negation elimination}$$

$$(\alpha \Rightarrow \beta) \equiv (\neg\beta \Rightarrow \neg\alpha) \quad \text{contraposition}$$

$$(\alpha \Rightarrow \beta) \equiv (\neg\alpha \vee \beta) \quad \text{implication elimination}$$

$$(\alpha \Leftrightarrow \beta) \equiv ((\alpha \Rightarrow \beta) \wedge (\beta \Rightarrow \alpha)) \quad \text{biconditional elimination}$$

$$\neg(\alpha \wedge \beta) \equiv (\neg\alpha \vee \neg\beta) \quad \text{De Morgan}$$

$$\neg(\alpha \vee \beta) \equiv (\neg\alpha \wedge \neg\beta) \quad \text{De Morgan}$$

$$(\alpha \wedge (\beta \vee \gamma)) \equiv ((\alpha \wedge \beta) \vee (\alpha \wedge \gamma)) \quad \text{distributivity of } \wedge \text{ over } \vee$$

$$(\alpha \vee (\beta \wedge \gamma)) \equiv ((\alpha \vee \beta) \wedge (\alpha \vee \gamma)) \quad \text{distributivity of } \vee \text{ over } \wedge$$

Валидност

- » Едно съждение е **валидно**, когато е вярно във **всички** модели
 - > Пример: $P \vee \neg P$
- » Нарича се **тавтология** и е безусловно вярно
 - > Всяко валидно съждение е **еквивалентно на true** – понеже true е вярно във всички модели
- » Защо се нуждаем от валидни съждения?
 - > От дефиницията можем да изведем теоремата за дедукция
 - > Позната още на старите гърци



Теорема за дедукция

- » За всички съждения α и β е в сила $\alpha \models \beta$ тогава, когато съждението $(\alpha \Rightarrow \beta)$ е валидно
- » Доказателство:
 - > И двете страни са еквивалентни на твърдението, че не съществува модел, в който α е вярно и β е невярно, т.е. няма модел, при който $\alpha \Rightarrow \beta$ е грешно
- » Следователно, можем да определим дали $\alpha \models \beta$ чрез проверка за това дали $(\alpha \Rightarrow \beta)$ е вярна във всеки модел (TT-ENTAILS?) или чрез доказване, че $(\alpha \Rightarrow \beta) \equiv \text{true}$
- » Обратно, теоремата за дедукцията гласи, че всяко валидно имплекативно съждение описва легитимен извод.



Удовлетвореност

- » Едно съждение е **удовлетворимо**, когато е вярно в **някой** модел
- » Пример: БЗ от предишния пример е удовлетворима, понеже е вярна за три модела.

$$R_1: \neg P_{1,1}$$

$$R_2: B_{1,1} \Leftrightarrow (P_{1,2} \vee P_{2,1})$$

$$R_3: B_{2,1} \Leftrightarrow (P_{1,1} \vee P_{2,1} \vee P_{3,1})$$

$$R_4: \neg B_{1,1}$$

$$R_5: B_{2,1}$$

$B_{1,1}$	$B_{2,1}$	$P_{1,1}$	$P_{1,2}$	$P_{2,1}$	$P_{2,2}$	$P_{3,1}$	R_1	R_2	R_3	R_4	R_5	KB
false	false	false	false	false	false	false	true	true	true	true	false	false
false	false	false	false	false	false	true	true	true	false	true	false	false
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮
false	true	false	false	false	false	false	true	true	false	true	true	false
false	true	false	false	false	true	false	true	true	true	true	true	true
false	true	false	false	false	true	true	true	true	true	true	true	true
false	true	false	false	false	false	true	true	true	true	true	true	<u>true</u>
false	true	false	false	false	true	true	true	true	true	true	true	<u>true</u>
false	true	false	false	true	false	false	true	false	false	true	true	false
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮
true	true	true	true	true	true	true	false	true	true	false	true	false

Валидност и удовлетвореност

- » Валидността и удовлетвореността са свързани
 - > α е валидно, когато $\neg\alpha$ не е удовлетворимо
- » Практически извод: $\alpha \models \beta$ тогава, когато $(\alpha \wedge \neg\beta)$ не е удовлетворимо
- » Доказателство (както преди):
 - > И двете страни са еквивалентни на твърдението, че не съществува модел, в който α е вярно и β е невярно



Валидност и удовлетвореност

- » Доказателството β от α посредством неудовлетворимост на $(\alpha \wedge \neg\beta)$ отговаря на познатия от математиката метод за **доказателство чрез противоречие**
 - > Изхождайки от това, че съждението β е грешно и показвайки, че това води до противоречие с позната аксиома α
 - > Това е точно, че $(\alpha \wedge \neg\beta)$ не е удовлетворимо



Правила за извод

» Правила (за извод), които могат да бъдат приложени за да се направи извод - верига от заключения

> Modus ponens

$$\frac{\alpha \Rightarrow \beta, \alpha}{\beta}$$

> И-Елиминирание

$$\frac{\alpha \wedge \beta}{\alpha}$$

> Всички логически еквивалентности от предишната таблица могат да се използват за правене на логически изводи



Примери

$(WumpusAhead \wedge WumpusAlive) \Rightarrow Shoot$

$(WumpusAhead \wedge WumpusAlive)$

Shoot

$(WumpusAhead \wedge WumpusAlive)$

WumpusAlive

Обобщение

- » Като се вземат предвид възможните стойности на истинността на α и β , може лесно да се покаже, че **Modus Ponens** и **And-Elimination** са винаги коректни
- » Тези правила могат да се използват във всеки конкретен случай, в който са приложими, за правене на коректни изводи
 - > **Без да е необходимо** да се използва методът “проверка на модели”



Обобщение

- » Всички **логически еквивалентности** могат да се използват също като правила за извод
- » Напр., еквивалентността на елиминирането на бикондиционала доставя две правила за извод

$$\frac{\alpha \Leftrightarrow \beta}{(\alpha \Rightarrow \beta) \wedge (\beta \Rightarrow \alpha)}$$

$$\frac{(\alpha \Rightarrow \beta) \wedge (\beta \Rightarrow \alpha)}{\alpha \Leftrightarrow \beta}$$



Обобщение

- » Не всички правила за извод работят и в двете посоки
- » Напр., не можем да изпълним Modus Ponens в обратна посока, за да получим $\alpha \Rightarrow \beta$ и α от β



Пример

$$R_1: \neg P_{1,1}$$

$$R_2: B_{1,1} \Leftrightarrow (P_{1,2} \vee P_{2,1})$$

$$R_3: B_{2,1} \Leftrightarrow (P_{1,1} \vee P_{2,2} \vee P_{3,1})$$

$$R_4: \neg B_{1,1}$$

$$R_5: B_{2,1}$$

Искаме да разберем липсата на дупка в $[1,2]$, т.е. да докажем $\neg P_{1,2}$

Първа стъпка:

Прилагаме biconditional elimination върху R_2

$$R_6: (B_{1,1} \Rightarrow (P_{1,2} \vee P_{2,1})) \wedge ((P_{1,2} \vee P_{2,1}) \Rightarrow B_{1,1})$$

Пример

$$R_1: \neg P_{1,1}$$

$$R_2: B_{1,1} \Leftrightarrow (P_{1,2} \vee P_{2,1})$$

$$R_3: B_{2,1} \Leftrightarrow (P_{1,1} \vee P_{2,2} \vee P_{3,1})$$

$$R_4: \neg B_{1,1}$$

$$R_5: B_{2,1}$$

$$R_6: (B_{1,1} \Rightarrow (P_{1,2} \vee P_{2,1})) \wedge ((P_{1,2} \vee P_{2,1}) \Rightarrow B_{1,1})$$

Искаме да разберем дали има дупка в [1,2], т.е. да докажем $\neg P_{1,2}$

Втора стъпка:

Прилагаме И-Елиминирание върху R_6

$$R_7: ((P_{1,2} \vee P_{2,1})) \Rightarrow B_{1,1}$$

Пример

$$R_1: \neg P_{1,1}$$

$$R_2: B_{1,1} \Leftrightarrow (P_{1,2} \vee P_{2,1})$$

$$R_3: B_{2,1} \Leftrightarrow (P_{1,1} \vee P_{2,2} \vee P_{3,1})$$

$$R_4: \neg B_{1,1}$$

$$R_5: B_{2,1}$$

$$R_6: (B_{1,1} \Rightarrow (P_{1,2} \vee P_{2,1})) \wedge ((P_{1,2} \vee P_{2,1}) \Rightarrow B_{1,1})$$

$$R_7: ((P_{1,2} \vee P_{2,1})) \Rightarrow B_{1,1}$$

Искаме да разберем дали има дупка в [1,2], т.е. да докажем $\neg P_{1,2}$

Трета стъпка:

Прилагаме контрапозиция върху R_7

$$R_8: (\neg B_{1,1} \Rightarrow \neg(P_{1,2} \vee P_{2,1}))$$

Пример

$$R_1: \neg P_{1,1}$$

$$R_2: B_{1,1} \Leftrightarrow (P_{1,2} \vee P_{2,1})$$

$$R_3: B_{2,1} \Leftrightarrow (P_{1,1} \vee P_{2,2} \vee P_{3,1})$$

$$R_4: \neg B_{1,1}$$

$$R_5: B_{2,1}$$

$$R_6: (B_{1,1} \Rightarrow (P_{1,2} \vee P_{2,1})) \wedge ((P_{1,2} \vee P_{2,1}) \Rightarrow B_{1,1})$$

$$R_7: ((P_{1,2} \vee P_{2,1})) \Rightarrow B_{1,1}$$

$$R_8: (\neg B_{1,1} \Rightarrow \neg(P_{1,2} \vee P_{2,1}))$$

Искаме да разберем дали има дупка в $[1,2]$, т.е. да докажем $\neg P_{1,2}$

Четвърта стъпка:

Прилагаме модус поненс върху R_8 и възприятието $R_4 (\neg B_{1,1})$

$$R_9: \neg(P_{1,2} \vee P_{2,1})$$

Пример

$R_1: \neg P_{1,1}$
 $R_2: B_{1,1} \Leftrightarrow (P_{1,2} \vee P_{2,1})$
 $R_3: B_{2,1} \Leftrightarrow (P_{1,1} \vee P_{2,2} \vee P_{3,1})$
 $R_4: \neg B_{1,1}$
 $R_5: B_{2,1}$
 $R_6: (B_{1,1} \Rightarrow (P_{1,2} \vee P_{2,1})) \wedge ((P_{1,2} \vee P_{2,1}) \Rightarrow B_{1,1})$
 $R_7: ((P_{1,2} \vee P_{2,1})) \Rightarrow B_{1,1}$
 $R_8: (\neg B_{1,1} \Rightarrow \neg(P_{1,2} \vee P_{2,1}))$
 $R_9: \neg(P_{1,2} \vee P_{2,1})$

Искаме да разберем дали има дупка в $[1,2]$, т.е. да докажем $\neg P_{1,2}$

Пета стъпка:

Прилагаме де Морган върху R_9

$R_{10}: \neg P_{1,2} \wedge \neg P_{2,1}$



Доказахме, че няма дупка в $[1,2]$ (също в $[2,1]$)

Доказателства

- » Преди направихме това доказателство на ръка.
- » Можем да приложим който и да е от алгоритмите за търсене за да намерим поредица от стъпки, които представляват доказателство.



Доказателства

- » При доказателствата трябва да дефинираме **един проблем**, както следва:
 - > **Начално състояние**: начална БЗ
 - > **Оператори**: правила за извод, прилагани към съждения, които съответстват на горната половина на правилото
 - > **Резултат от оператора**: добавяне съждение в долната половина на правилото за извод
 - > **Цел**: състояние, което съдържа съждението, което се опитваме да докажем



Доказателства

- » По този начин търсенето на доказателство е **алтернатива** на изброяване на модели
- » В много практически случаи намирането на доказателство може да бъде **по-ефективно**, защото доказателството може да пренебрегне несъществените предложения, независимо колко са



Пример

- » Разгледаното по-рано доказателство води до $\neg P_{1,2} \wedge \neg P_{2,1}$ без да разглежда съжденията $V_{2,1}$, $P_{1,1}$, $P_{2,2}$ или $P_{3,1}$
- » Те могат да бъдат **пренебрегнати**, защото **целевото съждение $P_{1,2}$ се появява само в съждението R_2** - другите съждения в R_2 се появяват само в R_4 и R_2
- » **Така, R_1 , R_3 и R_5 нямат отношение към доказателството**
- » Същото ще се случи, дори ако добавим още един милион съждения към БЗ
- » От друга страна, простият алгоритъм за таблица на истинността ще бъде претоварен от експоненциалната експлозия на моделите



Монотонност

- » Едно последно свойство на логическите системи е **МОНОТОННОСТТА**:
- » Множеството е изведените съждения може да се увеличава само когато информацията се добавя информацията към БЗ
- » За всеки две съждения α и β , ако $\text{БЗ} \models \alpha$, тогава $\text{БЗ} \wedge \beta \models \alpha$



Пример

- » Да предположим напр., че БЗ съдържа допълнително суждение β , в което се посочва, че в света на W . има точно 8 ями
- » Това знание може да помогне на агента да направи допълнителни заключения, но не може да обезсили извода, който вече е направен, като напр., че няма яма в $[1,2]$.
- » Монотонността означава, че правилата за изводи могат да се прилагат винаги, когато подходящите условия (премиси) се намират в БЗ
 - > Заключение на правилото трябва да следва **независимо от това какво друго е в БЗ**



Пълнота

- » Досега разгледаните правила са коректни
 - > Още не се занимаваме с проблема за пълнота на алгоритмите за извод, които ги използват
 - > Пълните алгоритми намират всички достъпни цели
- » Алгоритмите за търсене, като напр. итеративно търсене в дълбочина, са пълни в смисъл, че ще намерят всяка достижима цел
 - > Но ако наличните правила за извод са неадекватни, тогава целта не е достижима
- » Напр., ако премахнем правилото за елиминиране на бикондиционала, доказателството в предходния пример няма да успее



Резолюция

» Резолюция

- > Правило за извод, което комбинирано с произволен пълен алгоритъм за търсене, доставя пълен алгоритъм за извод



Единична резолюция

Единична резолюция:

$$\frac{\ell_1 \vee \dots \vee \ell_k, \quad m}{\ell_1 \vee \dots \vee \ell_{i-1} \vee \ell_{i+1} \vee \dots \vee \ell_k}$$

когато l литерал, а l_i и m **комплиментни** литерали

Единичната резолюция използва една клауза (дизюнкция от литерали) и един литерал, като от тях генерира нова клауза

Обобщена резолюция

Обобщена резолюция:

$$\frac{\ell_1 \vee \dots \vee \ell_k, \quad m_1 \vee \dots \vee m_n}{\ell_1 \vee \dots \vee \ell_{i-1} \vee \ell_{i+1} \vee \dots \vee \ell_k \vee m_1 \vee \dots \vee m_{j-1} \vee m_{j+1} \vee \dots \vee m_n}$$

когато ℓ_i и m_j са **комплиментни** литерали

Обобщената резолюция използва две клаузи, като от тях генерира една нова клауза

$$\frac{P_{1,1} \vee P_{3,1}, \quad \neg P_{1,1} \vee \neg P_{2,2}}{P_{3,1} \vee \neg P_{2,2}}$$

Пример

1,4	2,4	3,4	4,4
1,3 W!	2,3	3,3	4,3
1,2 S OK	2,2 OK	3,2	4,2
1,1 V OK	2,1 B V OK	3,1 P!	4,1

Агентът се връща обратно от [2,1] в [1,1] и след това продължава към [1,2], където възприема „миризма“, но не „полъх“

Пример

$$R_1: \neg P_{1,1}$$

$$R_2: B_{1,1} \Leftrightarrow (P_{1,2} \vee P_{2,1})$$

$$R_3: B_{2,1} \Leftrightarrow (P_{1,1} \vee P_{2,2} \vee P_{3,1})$$

$$R_4: \neg B_{1,1}$$

$$R_5: B_{2,1}$$

$$R_6: (B_{1,1} \Rightarrow (P_{1,2} \vee P_{2,1})) \wedge ((P_{1,2} \vee P_{2,1}) \Rightarrow B_{1,1})$$

$$R_7: ((P_{1,2} \vee P_{2,1}) \Rightarrow B_{1,1})$$

$$R_8: (\neg B_{1,1} \Rightarrow \neg(P_{1,2} \vee P_{2,1}))$$

$$R_9: \neg(P_{1,2} \vee P_{2,1})$$

$$R_{10}: \neg P_{1,2} \wedge \neg P_{2,1}$$

Добавяме към базата знания следните правила:

$$R_{11}: \neg B_{1,2}$$

$$R_{12}: B_{1,2} \Leftrightarrow (P_{1,1} \vee P_{2,2} \vee P_{1,3})$$

Пример

$$R_1: \neg P_{1,1}$$

$$R_2: B_{1,1} \Leftrightarrow (P_{1,2} \vee P_{2,1})$$

$$R_3: B_{2,1} \Leftrightarrow (P_{1,1} \vee P_{2,2} \vee P_{3,1})$$

$$R_4: \neg B_{1,1}$$

$$R_5: B_{2,1}$$

$$R_6: (B_{1,1} \Rightarrow (P_{1,2} \vee P_{2,1})) \wedge ((P_{1,2} \vee P_{2,1}) \Rightarrow B_{1,1})$$

$$R_7: ((P_{1,2} \vee P_{2,1}) \Rightarrow B_{1,1})$$

$$R_8: (\neg B_{1,1} \Rightarrow \neg(P_{1,2} \vee P_{2,1}))$$

$$R_9: \neg(P_{1,2} \vee P_{2,1})$$

$$R_{10}: \neg P_{1,2} \wedge \neg P_{2,1}$$

$$R_{11}: \neg B_{1,2}$$

$$R_{12}: B_{1,2} \Leftrightarrow (P_{1,1} \vee P_{2,2} \vee P_{1,3})$$

По същия начин както преди
можем да изведем липсата
на дупки в $[2,2]$ и $[1,3]$:

$$R_{13}: \neg P_{2,2}$$

$$R_{14}: \neg P_{1,3}$$

Пример

$$R_1: \neg P_{1,1}$$

$$R_2: B_{1,1} \Leftrightarrow (P_{1,2} \vee P_{2,1})$$

$$R_3: B_{2,1} \Leftrightarrow (P_{1,1} \vee P_{2,2} \vee P_{3,1})$$

$$R_4: \neg B_{1,1}$$

$$R_5: B_{2,1}$$

$$R_6: (B_{1,1} \Rightarrow (P_{1,2} \vee P_{2,1})) \wedge ((P_{1,2} \vee P_{2,1}) \Rightarrow B_{1,1})$$

$$R_7: ((P_{1,2} \vee P_{2,1}) \Rightarrow B_{1,1})$$

$$R_8: (\neg B_{1,1} \Rightarrow \neg(P_{1,2} \vee P_{2,1}))$$

$$R_9: \neg(P_{1,2} \vee P_{2,1})$$

$$R_{10}: \neg P_{1,2} \wedge \neg P_{2,1}$$

$$R_{11}: \neg B_{1,2}$$

$$R_{12}: B_{1,2} \Leftrightarrow (P_{1,1} \vee P_{2,2} \vee P_{1,3})$$

$$R_{13}: \neg P_{2,2}$$

$$R_{14}: \neg P_{1,3}$$

Освен това при прилагане на *biconditional elimination*, последван от модус поненс за R_3 извеждаме, че има дупка в $[1,1]$, $[2,2]$ или $[3,1]$:

$$R_{15}: P_{1,1} \vee P_{2,2} \vee P_{3,1}$$

Пример

$R_1: \neg P_{1,1}$
 $R_2: B_{1,1} \Leftrightarrow (P_{1,2} \vee P_{2,1})$
 $R_3: B_{2,1} \Leftrightarrow (P_{1,1} \vee P_{2,2} \vee P_{3,1})$
 $R_4: \neg B_{1,1}$
 $R_5: B_{2,1}$
 $R_6: (B_{1,1} \Rightarrow (P_{1,2} \vee P_{2,1})) \wedge ((P_{1,2} \vee P_{2,1}) \Rightarrow B_{1,1})$
 $R_7: ((P_{1,2} \vee P_{2,1}) \Rightarrow B_{1,1})$
 $R_8: (\neg B_{1,1} \Rightarrow \neg(P_{1,2} \vee P_{2,1}))$
 $R_9: \neg(P_{1,2} \vee P_{2,1})$
 $R_{10}: \neg P_{1,2} \wedge \neg P_{2,1}$
 $R_{11}: \neg B_{1,2}$
 $R_{12}: B_{1,2} \Leftrightarrow (P_{1,1} \vee P_{2,2} \vee P_{3,1})$
 $R_{13}: \neg P_{2,2}$
 $R_{14}: \neg P_{1,3}$
 $R_{15}: P_{1,1} \vee P_{2,2} \vee P_{3,1}$

Сега идва първото прилагане на правилото на резолюцията: литералът $\neg P_{2,2}$ в R_{13} резюлира с литерала $P_{2,2}$ в R_{15} и се генерира резолвента:

$$R_{16}: P_{1,1} \vee P_{3,1}$$

Т.е., когато има дупка в едно от полетата $[1,1]$, $[2,2]$ или $[3,1]$ и тя не е в $[2,2]$, тогава тя се намира в $[1,1]$ или $[3,1]$

Пример

$R_1: \neg P_{1,1}$
 $R_2: B_{1,1} \Leftrightarrow (P_{1,2} \vee P_{2,1})$
 $R_3: B_{2,1} \Leftrightarrow (P_{1,1} \vee P_{2,2} \vee P_{3,1})$
 $R_4: \neg B_{1,1}$
 $R_5: B_{2,1}$
 $R_6: (B_{1,1} \Rightarrow (P_{1,2} \vee P_{2,1})) \wedge ((P_{1,2} \vee P_{2,1}) \Rightarrow B_{1,1})$
 $R_7: ((P_{1,2} \vee P_{2,1}) \Rightarrow B_{1,1})$
 $R_8: (\neg B_{1,1} \Rightarrow \neg(P_{1,2} \vee P_{2,1}))$
 $R_9: \neg(P_{1,2} \vee P_{2,1})$
 $R_{10}: \neg P_{1,2} \wedge \neg P_{2,1}$
 $R_{11}: \neg B_{1,2}$
 $R_{12}: B_{1,2} \Leftrightarrow (P_{1,1} \vee P_{2,2} \vee P_{1,3})$
 $R_{13}: \neg P_{2,2}$
 $R_{14}: \neg P_{1,3}$
 $R_{15}: P_{1,1} \vee P_{2,2} \vee P_{3,1}$
 $R_{16}: P_{1,1} \vee P_{3,1}$

Освен това литералът $\neg P_{1,1}$ в R_1 резюлира с литерала $P_{1,1}$ в R_{16} и се генерира резолвента:

$$R_{17}: P_{3,1}$$

Т.е., когато има дупка в едно от полетата $[1,1]$ или $[3,1]$ и тя не е в $[1,1]$, тогава тя се намира в $[3,1]$

КНФ

- » Правилото за резолюцията се прилага само за клаузи (т.е. дизюнкции на литерали), така че изглежда, че е приложимо само за бази знания и заявки, състоящи се от клаузи.
- » Как тогава може да доведе до пълна процедура за извод за цялата пропозиционална логика?
- » Отговорът е, че **всяко съждение** от съждителната логика е логически **еквивалентно на конюнкция от клаузи**.
- » За съждение, представено като конюнкция от клаузи, се казва, че е в **конюнктивна нормална форма (КНФ)**. Сега описваме процедура за конвертиране в CNF.

Преобразуване в КНФ

Ние илюстрираме процедурата, като преобразуваме съждението $B_{1,1} \Leftrightarrow (P_{1,2} \vee P_{2,1})$ в CNF. Стъпките са както следва:

- 1 Елиминираме \Leftrightarrow , заменяйки $\alpha \Leftrightarrow \beta$ с $(\alpha \Rightarrow \beta) \wedge (\beta \Rightarrow \alpha)$:

$$(B_{1,1} \Rightarrow (P_{1,2} \vee P_{2,1})) \wedge ((P_{1,2} \vee P_{2,1}) \Rightarrow B_{1,1}) .$$

- 2 Елиминираме \Rightarrow , заменяйки $\alpha \Rightarrow \beta$ с $\neg\alpha \vee \beta$:

$$(\neg B_{1,1} \vee P_{1,2} \vee P_{2,1}) \wedge (\neg(P_{1,2} \vee P_{2,1}) \vee B_{1,1}) .$$

Преобразуване в КНФ

- 3 CNF изисква \neg да се появява само в литерали, така че въведем \neg „вътре“ чрез многократно прилагане на следните еквивалентности:

$$\neg(\neg\alpha) \equiv \alpha \quad (\text{double-negation elimination})$$

$$\neg(\alpha \wedge \beta) \equiv (\neg\alpha \vee \neg\beta) \quad (\text{De Morgan})$$

$$\neg(\alpha \vee \beta) \equiv (\neg\alpha \wedge \neg\beta) \quad (\text{De Morgan})$$

- 4 В примера изискваме само едно приложение на последното правило:

$$(\neg B_{1,1} \vee P_{1,2} \vee P_{2,1}) \wedge ((\neg P_{1,2} \wedge \neg P_{2,1}) \vee B_{1,1}) .$$

- 5 Сега имаме съждение, съдържащо вложени оператори \wedge и \vee , приложени към литерали. Прилагаме дистрибутивния закон, разпределяйки \vee върху \wedge , където е възможно.

$$(\neg B_{1,1} \vee P_{1,2} \vee P_{2,1}) \wedge (\neg P_{1,2} \vee B_{1,1}) \wedge (\neg P_{2,1} \vee B_{1,1}) .$$

Резолюционен алгоритъм

```
function PL-RESOLUTION( $KB, \alpha$ ) returns true or false
  inputs:  $KB$ , the knowledge base, a sentence in propositional logic
            $\alpha$ , the query, a sentence in propositional logic

   $clauses \leftarrow$  the set of clauses in the CNF representation of  $KB \wedge \neg\alpha$ 
   $new \leftarrow \{ \}$ 
  loop do
    for each pair of clauses  $C_i, C_j$  in  $clauses$  do
       $resolvents \leftarrow$  PL-RESOLVE( $C_i, C_j$ )
      if  $resolvents$  contains the empty clause then return true
       $new \leftarrow new \cup resolvents$ 
  if  $new \subseteq clauses$  then return false
   $clauses \leftarrow clauses \cup new$ 
```

Коментар

- » Процедурите за извод, базирани на резолюцията, работят чрез използване на принципа на доказателство чрез противоречие.
- » Тоест, за да покажем, че $KV \models \alpha$, ние показваме, че $(KV \wedge \neg\alpha)$ е неудовлетворимо.
- » Правим това, като докажем противоречие.

Коментар

- » Алгоритъмът за резолюцията работи по следната схема:
- » Първо ($KB \wedge \neg\alpha$) се преобразува в CNF.
- » След това върху получените клаузи се прилага правилото на резолюцията.
- » Всяка двойка, която съдържа резолюиращи се литерали, създава нова клауза, която се добавя към съществуващото множество, ако вече не е налична.
- » Процесът продължава, докато се случи едно от двете неща:
 - > Няма нови клаузи, които могат да бъдат добавени, в който случай KB не води до α ;
 - > Две клаузи се резюлират, за да дадат празната клауза, в който случай KB води до α .

Коментар

- » Празната клауза — дизюнкция без дизюнкти — е еквивалентна на False, защото дизюнкцията е вярна само ако поне един от нейните дизюнкти е верен.
- » Друг начин да видим, че празна клауза представлява противоречие, е да забележим, че тя възниква само от резолюцията на две допълващи се единични клаузи като P и $\neg P$.

Пример

Можем да приложим процедурата за разделяне към много просто заключение в света на wumpus. Когато агентът е в [1,1] и няма полъх – т.е. не може да има ями в съседните полета.

Съответната база знания е:

$$\checkmark \text{ KB} = R_2 \wedge R_4 = (B_{1,1} \Leftrightarrow (P_{1,2} \vee P_{2,1})) \wedge \neg B_{1,1}$$

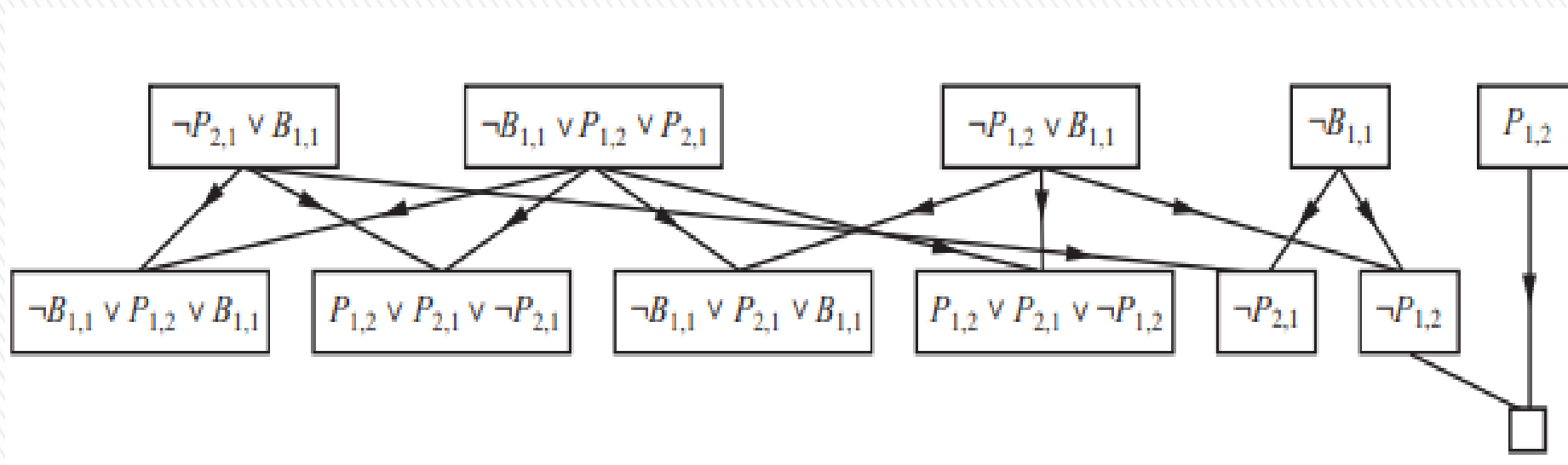
Искаме да докажем $\alpha = \neg P_{1,2}$.

Пример

Когато преобразуваме $(KB \wedge \neg\alpha)$ в CNF, получаваме клаузите, показани в горната част на фигурата.

Вторият ред на фигурата показва клаузи, получени чрез разрешаване на двойки в първия ред.

След това, когато $P_{1,2}$ разолюира с $\neg P_{1,2}$, получаваме празната клауза, показана като малък квадрат.



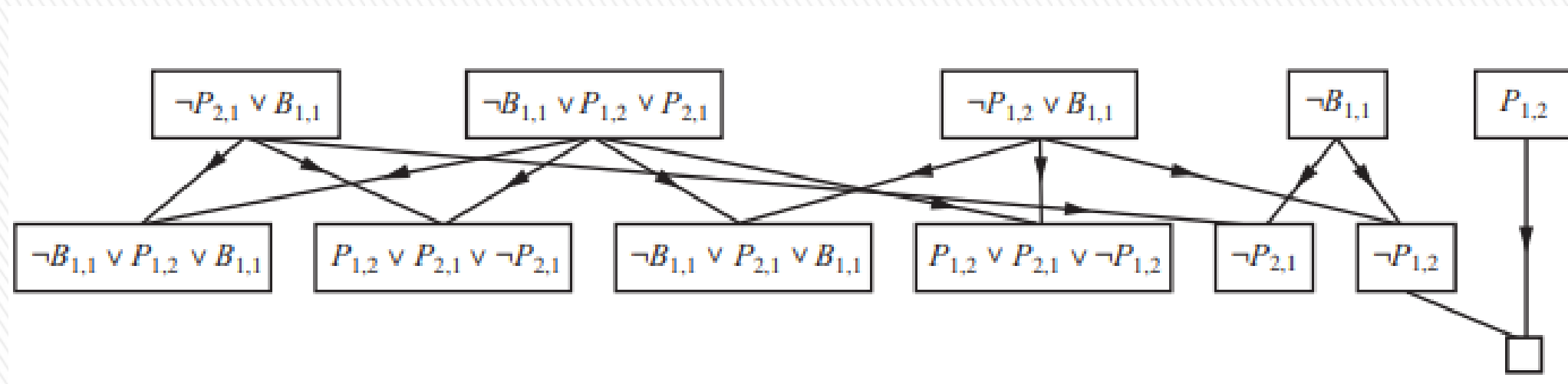
Пример

Една проверка разкрива, че много стъпки са безсмислени.

Например клаузата $B_{1,1} \vee \neg B_{1,1} \vee P_{1,2}$ е еквивалентна на $\text{True} \vee P_{1,2}$, което от сво страна е еквивалентно на True .

Изводът, че True е вярно, не е много полезен.

Следователно всяка клауза, в която се появяват два резолюиращи се литерала, може да бъде отхвърлена.





Благодаря за вниманието!