

Satslogik – grundläggande definitioner

- Satslogikens syntax (välformade formler)
- Satslogikens semantik (tolkningar)
- Modeller, logisk konsekvens och ekvivalens
- Några notationella förenklingar
- Kompletta mängder av konnektiv

Satslogik

Definition En **satslogik** är ett system $L = (\mathbf{P}, \mathbf{C}, \mathbf{A})$ där

- \mathbf{P} är en mängd av symboler som kallas **atomer** (*proposition names*),
- $\mathbf{C} \subseteq \{\neg, \rightarrow, \wedge, \vee, \leftrightarrow, \perp, \top\}$ är mängden av **konnektiv** (*connectives*),
- $\mathbf{A} = \{(\cdot)\}$ är mängden av **hjälpssymboler** (*auxiliary symbols*) och
- $\mathbf{P} \cap (\mathbf{C} \cup \mathbf{A}) = \emptyset$.

Anmärkingar

- Normalt används versaler $A, B, C, A', A_2, B_0, \dots$ som atomer.
- Blockvärldsexemplet är en satslogik om de atomära satserna $\text{On_table}(K_1)$ osv betraktas som en symbol var (snarare än som sammansatta uttryck). Logiken sägs då vara en **ändlig** satslogik eftersom antalet atomer är ändligt.

Uppgift 1

Vilka av följande kan sägas vara satslogiska system enligt definitionen på föregående sida?

- $L_1 = (\{A, B, C\}, \{\neg, \rightarrow, \wedge, \vee, \leftrightarrow, \perp, \top\}, \{(\cdot)\})$
- $L_2 = (\{\wedge, \vee\}, \{\neg, \wedge, \vee\}, \{(\cdot)\})$
- $L_3 = (\{\otimes, \oplus\}, \{\perp, \top\}, \{(\cdot)\})$
- $L_4 = (\{\wedge, \vee\}, \{\neg, \rightarrow\}, \{(\cdot)\})$
- $L_5 = (\{A, B, C\}, \{\neg, \rightarrow, \wedge, \vee, \leftrightarrow, \perp, \top\}, \{(\cdot), \emptyset\})$
- $L_6 = (\{A, B, C\}, \{\neg, \vee\}, \{(\cdot)\})$

Lösning

Enligt definitionen är L_3, L_4 och L_6 satslogiska system. Men

- L_1 använder sig av hakparenteser som hjälpssymboler, trots att definition kräver vanliga parenteser. För att undvika missförstånd bör man följa definitionen, även om den ibland kan tyckas onödigt begränsande. Vidare,
- $L_2 = (\mathbf{P}, \mathbf{C}, \mathbf{A})$ har $\mathbf{P} \cap (\mathbf{C} \cup \mathbf{A}) \neq \emptyset$ och
- $L_5 = (\mathbf{P}, \mathbf{C}, \mathbf{A}, \emptyset)$ har en extra komponent.

Dessa system strider alltså mot definition.

Välformade formler – en induktiv definition

Definition Låt $L = (\mathbf{P}, \mathbf{C}, \mathbf{A})$ vara en satslogik. Mängden av alla **välformade formler över L** är den minsta mängden $\text{WF}(L) \subseteq (\mathbf{P} \cup \mathbf{C} \cup \mathbf{A})^*$ sådan att

- $\mathbf{P} \subseteq \text{WF}(L)$ (varje atom är en välformad formel),
- $\perp \in \text{WF}(L)$ och $\top \in \text{WF}(L)$, förutsatt att \perp respektive \top finns i \mathbf{C} ,
- $(\neg\varphi) \in \text{WF}(L)$ för varje $\varphi \in \text{WF}(L)$, förutsatt att \neg finns i \mathbf{C} och
- $(\varphi \otimes \varphi') \in \text{WF}(L)$ för alla $\varphi, \varphi' \in \text{WF}(L)$ och varje $\otimes \in \{\rightarrow, \wedge, \vee, \leftrightarrow\}$ som finns i \mathbf{C} .

Anmärkingar

- Om inget annat är sagt antas att $\mathbf{C} = \{\neg, \rightarrow, \wedge, \vee, \leftrightarrow, \perp, \top\}$.
- Om $\mathbf{P} = \{A, B, C\}$ så är $((\neg A) \vee (B \wedge \perp))$ och $(A \rightarrow (\neg(A \vee C)))$ välformade men varken $\neg A \vee (B \wedge \perp)$ eller $\neg \neg A$ – och absolut inte $\forall A \neg \perp B \rightarrow$.
- De strikta reglerna för att sätta parenteser ska snart mjukas upp...

Uppgift 2

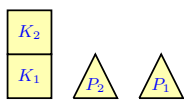
Vilka av följande formler är välformade enligt definitionen på föregående sida?

$$\begin{array}{ll} \varphi_1 = \perp & \varphi_5 = () \\ \varphi_2 = \neg A & \varphi_6 = (\neg A) \\ \varphi_3 = (\neg(A \wedge B)) & \varphi_7 = (\rightarrow (A \wedge B)) \\ \varphi_4 = (A \vee B \vee C) & \varphi_8 = ((A \vee B) \vee C) \end{array}$$

Lösning

Enligt definitionen är $\varphi_1, \varphi_3, \varphi_6$ och φ_8 välformade formler, men

- φ_2 saknar parenteser, och det gör även
- φ_4 (antingen runt $A \vee B$ eller runt $A \vee C$). Dessutom,
- φ_5 består bara av parenteser, och
- φ_6 ignorerar att ranken av \rightarrow är två.

I blockvärldsexemplet definieras  så här:

Atom A	v(A)	Atom A	v(A)	Atom A	v(A)	Atom A	v(A)
Is_cube(K ₁)	1	Is_pyramid(P ₂)	1	On(K ₁ , P ₁)	0	On(P ₁ , K ₂)	0
Is_cube(K ₂)	1	On_table(K ₁)	1	On(K ₁ , P ₂)	0	On(P ₁ , P ₁)	0
Is_cube(P ₁)	0	On_table(K ₂)	0	On(K ₂ , K ₁)	1	On(P ₁ , P ₂)	0
Is_cube(P ₂)	0	On_table(P ₁)	1	On(K ₂ , K ₂)	0	On(P ₂ , K ₁)	0
Is_pyramid(K ₁)	0	On_table(P ₂)	1	On(K ₂ , P ₁)	0	On(P ₂ , K ₂)	0
Is_pyramid(K ₂)	0	On(K ₁ , K ₁)	0	On(K ₂ , P ₂)	0	On(P ₂ , P ₁)	0
Is_pyramid(P ₁)	1	On(K ₁ , K ₂)	0	On(P ₁ , K ₁)	0	On(P ₂ , P ₂)	0

Antalet möjliga tolkningar är 2^{28} fastän bara 13 representerar "realiteten". Stryks de atomer som har samma värde i **alla** tillåtna världar så reduceras antalet atomer till 10, motsvarande 1024 tolkningar, vilket är mycket bättre!

Semantik 3 – formell definition av formlers semantik

Definition En tolkning $v: \mathbf{P} \rightarrow \{0, 1\}$ utvidgas induktivt till en funktion $\bar{v}: \text{WF}(\mathbf{L}) \rightarrow \{0, 1\}$: För $\varphi \in \text{WF}(\mathbf{L})$

$$\bar{v}(\varphi) = \begin{cases} \min(\bar{v}(\varphi_1), \bar{v}(\varphi_2)) & \text{om } \varphi = (\varphi_1 \wedge \varphi_2) \\ \max(\bar{v}(\varphi_1), \bar{v}(\varphi_2)) & \text{om } \varphi = (\varphi_1 \vee \varphi_2) \\ \max(1 - \bar{v}(\varphi_1), \bar{v}(\varphi_2)) & \text{om } \varphi = (\varphi_1 \rightarrow \varphi_2) \\ 1 - (\bar{v}(\varphi_1) - \bar{v}(\varphi_2))^2 & \text{om } \varphi = (\varphi_1 \leftrightarrow \varphi_2) \\ 1 - \bar{v}(\varphi_1) & \text{om } \varphi = (\neg \varphi_1) \\ 0 & \text{om } \varphi = \perp \\ 1 & \text{om } \varphi = \top \\ v(\varphi) & \text{om } \varphi \in \mathbf{P}. \end{cases}$$

Semantik 1 – sanningsvärden och tolkningar

- **Sanningsvärdena** (*truth values*) är 0 och 1. De står för **falskt** och **sant** vilket gör att det i litteraturen ofta används F och T istället.
- En **tolkning** (*interpretation, valuation*) är en funktion

$$v: \mathbf{P} \rightarrow \{0, 1\}.$$

Den tilldelar alltså varje atom ett sanningsvärde och definierar således en (ev inte tillåten) värld.

Observation: Även i en ändlig satslogik finns det rätt många tolkningar, nämligen 2^n stycken där n är antalet element i \mathbf{P} .

Semantik 2 – informell definition av formlers semantik

Semantiken för konnektiv (och därmed för formler) kan beskrivas med **sanningstabeller**. För \neg och \rightarrow ser det så här ut (där $\varphi, \varphi' \in \text{WF}(\mathbf{L})$):

v(φ)	v(¬φ)
0	1
1	0

v(φ)	v(φ')	v((φ → φ'))
0	0	1
0	1	1
1	0	0
1	1	1

Det brukar vara svårt att inse att $(\varphi \rightarrow \varphi')$ är sann om φ är falsk, oberoende av φ' s värde. Det blir dock enkelt om man kommer ihåg att det som är sant är **påståendet som en helhet**. Med andra ord, formeln hävdar endast att φ' är sann **under förutsättningen** att φ är sann. Om förutsättningen inte är given säger formeln alltså inte fel.

Ett alternativt sätt att säga samma sak:

Definition En tolkning $v: \mathbf{P} \rightarrow \{0, 1\}$ utvidgas induktivt till en funktion $\bar{v}: \text{WF}(\mathbf{L}) \rightarrow \{0, 1\}$ så här:

$$\begin{aligned} \bar{v}((\varphi_1 \wedge \varphi_2)) = 1 & \quad \text{om och endast om} \quad \bar{v}(\varphi_1) = 1 \text{ och } \bar{v}(\varphi_2) = 1, \\ \bar{v}((\varphi_1 \vee \varphi_2)) = 1 & \quad \text{om och endast om} \quad \bar{v}(\varphi_1) = 1 \text{ eller } \bar{v}(\varphi_2) = 1, \\ \bar{v}((\varphi_1 \rightarrow \varphi_2)) = 1 & \quad \text{om och endast om} \quad \bar{v}(\varphi_1) = 0 \text{ eller } \bar{v}(\varphi_2) = 1, \\ \bar{v}((\varphi_1 \leftrightarrow \varphi_2)) = 1 & \quad \text{om och endast om} \quad \bar{v}(\varphi_1) = \bar{v}(\varphi_2), \\ \bar{v}((\neg \varphi)) = 1 & \quad \text{om och endast om} \quad \bar{v}(\varphi) = 0, \\ \bar{v}(\perp) = 0, & \\ \bar{v}(\top) = 1 & \quad \text{och} \\ \bar{v}(A) = v(A) & \quad \text{för alla } A \in \mathbf{P}. \end{aligned}$$

Uppgift 3

Låt $v(A) = 0$ och $v(B) = 1$, vad är då $\bar{v}((\neg A) \rightarrow (A \vee B))$?

Modell och logisk följd

Nedan, låt $\Phi \subseteq \text{WF}(\mathbf{L})$ vara en mängd av formler och $\varphi \in \text{WF}(\mathbf{L})$ en formel.

- En **modell** av Φ är en tolkning v sådan att $\bar{v}(\varphi) = 1$ för alla $\varphi \in \Phi$.
Intuitivt: En modell är en situation där alla formler i Φ är sanna.
- Mängden av alla modeller av Φ skrivs $\text{Mod}(\Phi)$. Istället för $\text{Mod}(\{\varphi\})$ kan kort skrivas $\text{Mod}(\varphi)$.
- Vi säger att φ är en **logisk följd** (också kallad **semantisk följd**) av Φ och skriver $\Phi \models \varphi$ om $\text{Mod}(\Phi) \subseteq \text{Mod}(\varphi)$.
Intuitivt: Om formlerna i Φ är sanna i en situation så är φ också sann i samma situation.
- Istället för $\emptyset \models \varphi$ kan kort skrivas $\models \varphi$. (Observera att $\text{Mod}(\emptyset)$ är mängden av alla tolkningar!)
- Formeln φ är **satisfierbar** om $\text{Mod}(\varphi) \neq \emptyset$. Den är en **tautologi** om varje tolkning är en modell av φ (alltså om $\models \varphi$).

Logisk ekvivalens: Formler $\varphi, \varphi' \in \text{WF}(\mathbf{L})$ är **logiskt ekvivalenta** och vi skriver $\varphi \equiv \varphi'$ om $\text{Mod}(\varphi) = \text{Mod}(\varphi')$. Några viktiga ekvivalenser:

$(\varphi \wedge \perp) \equiv \perp$	\perp är nollan för \wedge
$(\varphi \vee \perp) \equiv \varphi$	\perp är ettan för \vee
$(\varphi \wedge (\neg\varphi)) \equiv \perp$	\wedge -komplementregel
$(\varphi \wedge \varphi) \equiv \varphi$	idempotens
$((\varphi \wedge \varphi') \wedge \varphi'') \equiv (\varphi \wedge (\varphi' \wedge \varphi''))$	associativitet av \wedge
$(\varphi \wedge \varphi') \equiv (\varphi' \wedge \varphi)$	kommutativitet av \wedge
$(\varphi \wedge (\varphi' \vee \varphi'')) \equiv ((\varphi \wedge \varphi') \vee (\varphi \wedge \varphi''))$	\wedge distribuerar över \vee
$(\varphi \wedge (\varphi \vee \varphi')) \equiv \varphi$	absorptionsregel
$(\neg(\varphi \wedge \varphi')) \equiv ((\neg\varphi) \vee (\neg\varphi'))$	de Morgans regel

För varje ekvivalens fås en **dual** ekvivalens genom att byta ut varje konnektiv mot dess duala: $\wedge \longleftrightarrow \vee$, $\perp \longleftrightarrow \top$, $\leftrightarrow \longleftrightarrow \leftrightarrow$, $\neg \longleftrightarrow \neg$. På så sätt fås t ex de Morgans andra regel: $(\neg(\varphi \vee \varphi')) \equiv ((\neg\varphi) \wedge (\neg\varphi'))$.

Lösning

$$\begin{aligned}
 \bar{v}((\neg A) \rightarrow (A \vee B)) &= \max(1 - \bar{v}(\neg A), \bar{v}(A \vee B)) \\
 &= \max(1 - (1 - \bar{v}(A)), \max(\bar{v}(A), \bar{v}(B))) \\
 &= \max(1 - (1 - v(A)), \max(v(A), v(B))) \\
 &= \max(1 - (1 - 0), \max(0, 1)) \\
 &= \max(1 - 1, 1) \\
 &= 1
 \end{aligned}$$

Exempel: Låt v, v' vara tolkningarna som motsvarar



Både v och v' är modeller av

$$\Phi = \{(\text{On}(P_1, K_2) \rightarrow \text{On}(P_2, K_1)), (\text{On}(P_1, K_1) \vee \text{On}(P_1, K_2))\},$$

dvs $v, v' \in \text{Mod}(\Phi)$. Båda formler är således satisfierbara. Ingen av dem är dock en tautologi eftersom det finns tolkningar som gör att de blir falska. Dessutom gäller t ex $\Phi \models (\text{On}(P_1, K_1) \vee \text{On}(P_2, K_1))$.

Frågor:

- Är $(\text{On_table}(P_1) \rightarrow (\neg \text{On}(P_1, K_1)))$ en tautologi (alltså alltid sann)?
- Är $\text{Is_pyramid}(K_1)$ satisfierbar (alltså ibland sann)?
- Gäller $\{\text{On_table}(K_2)\} \models (\neg \text{On}(K_2, K_1))$?

Förenklad notation

- Pga associativitet kan $((\varphi \wedge \varphi') \wedge \varphi'')$ och $(\varphi \wedge (\varphi' \wedge \varphi''))$ uppfattas som samma formel och skrivas $(\varphi \wedge \varphi' \wedge \varphi'')$. Samma sak gäller \vee .
- Pga kommutativitet kan $(\varphi \wedge \varphi')$ och $(\varphi' \wedge \varphi)$ uppfattas som samma formel. Samma sak gäller \vee .
- Parenteser runt negerade atomer samt yttre parenteser kan utelämnas.
- I litteraturen används dessutom ofta bindningsregler: \neg före \wedge före \vee före \rightarrow och \leftrightarrow . För att undvika missförstånd använder vi inte de här reglerna!

Kompleta mängder av konnektiv

Alla tänkbara logiska konnektiv kan översättas till ekvivalenta formler som endast inne- håller konnektiven \rightarrow och \neg (och ev \perp). Konnektivmängden $\{\rightarrow, \neg\}$ kallas därför **komplett**.

Ytterligare kompletta mängder av konnektiv är t ex $\{\wedge, \neg\}$ och $\{\vee, \neg\}$. Det finns dock två konnektiv som bildar kompletta mängder för sig själva:

NAND (not and)			NOR (not or)		
$\overline{\varphi}$	$\overline{\varphi'}$	$\overline{\varphi \wedge \varphi'}$	$\overline{\varphi}$	$\overline{\varphi'}$	$\overline{\varphi \vee \varphi'}$
0	0	1	0	0	1
0	1	1	0	1	0
1	0	1	1	0	0
1	1	0	1	1	0

Med andra ord, om vi istället för $\neg, \rightarrow, \wedge, \vee, \leftrightarrow$ endast har $|$ eller \downarrow kan vi fortfarande uttrycka lika mycket.