1.6 Semantiska träd

Betrakta följande exempel

Exempel 30. Vi undersöker om formeln $(p_0 \wedge ((p_0 \to p_1) \wedge ((p_1 \to p_2) \wedge ((p_2 \to p_3) \wedge \neg p_3))))$ är satisfierbar.

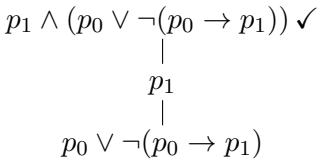
Vi kan alltid konstruera sanningsvärdetabellen för formeln men med fyra satssymboler blir den obehändigt stor. Istället lönar det sig att se på delformlerna och vilka krav de ställer på en värdering som skulle göra formeln sann. Genom att bit för bit bryta ner konjunktionerna ser vi att varje värdering v som satisfierar formeln också måste uppfylla

$$v(p_0) = v(p_0 \to p_1) = v(p_1 \to p_2) = v(p_2 \to p_3) = v(\neg p_3) = 1$$

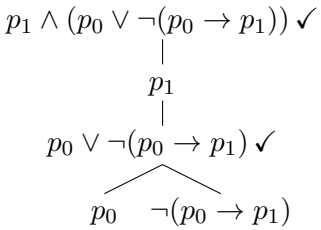
Eftersom $v(p_0) = 1$ och $v(p_0 \to p_1) = 1$, måste $v(p_1) = 1$. På samma sätt ser vi att $v(p_2) = 1$ och vidare att $v(p_3) = 1$. Samtidigt har vi kravet att $v(\neg p_3) = 1$ vilket är omöjligt. Formeln är alltså inte satisfierbar.

Resonemanget ovan kan generaliseras till en systematisk metod, kallad trädmetoden eller tablåmetoden. I den undersöker vi med hjälp av ett *semantiskt träd* vilka delformler som måste vara sanna för att formeln skall vara sann. Antag att vi vill avgöra om formeln $A = p_1 \wedge (p_0 \vee \neg(p_0 \to p_1))$ är satisfierbar. Vi konstruerar då ett semantiskt träd med formeln A överst och "bryter ner" formeln ett konnektiv i taget, tills bara literaler (satssymboler eller negerade satssymboler) återstår. Nedbrytningsreglerna återspeglar sanningskraven för en sammansatt formel: måste endera eller båda delformlerna vara sanna för att formeln skall vara sann?

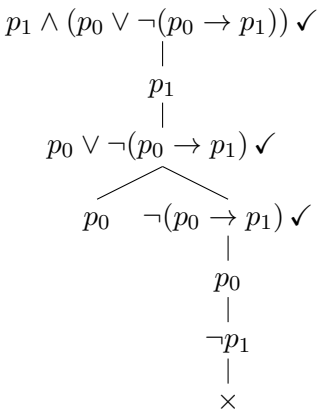
Formeln A är en konjunktion så för att den skall vara sann måste båda delformlerna p_1 och $(p_0 \vee \neg(p_0 \to p_1))$ vara sanna.



Aneftersom formlerna bryts ner markerar vi dem med ✓. I nästa steg ser vi att $p_0 \vee \neg(p_0 \to p_1)$ är en disjunktion. För att den skall vara sann måste endera disjunkten vara sann. Trädet förgrenar sig:

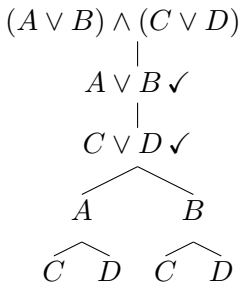


I nästa steg bryter vi ner formeln $\neg(p_0 \to p_1)$. Den är en negation av en implikation. Den stämmer om förledet är sant och efterledet falskt.



När alla formler som inte är literaler är markerade med ✓ är trädet färdigt. Om någon gren innehåller både en formel och dess negation markeras detta med ett kryss i slutet på grenen. Grenen kallas då *sluten*. De övriga grenarna är *öppna*. Varje öppen gren ger upphov till en värdering som gör alla formler på grenen sanna, inkuluive den översta. Vi ser alltså från trädet ovan att värderingar som uppfyller $v(p_0) = v(p_1) = 1$ satisfierar A.

Nedbrytningsreglerna för semantiska träd finns sammanställda i Tabell 3. Då ett träd förgrenar sig skall reglerna tillämpas på alla öppna grenar under formeln. Således ger formeln $(A \vee B) \wedge (C \vee D)$ upphov till ett träd med fyra grenar:



Konjunktion	$A \wedge B$ A B	$\neg(A \wedge B)$ $\neg A$ $\neg B$	
Disjunktion	$A \vee B$ A B	$\neg(A \vee B)$ $\neg A$ $\neg B$	
Negation	$\neg\neg A$ A	A $\neg A$ ×	$\neg A$ A ×

Implikation	$A \to B$ $\neg A$ B	$\neg(A \to B)$ A $\neg B$
Ekvivalens	$A \leftrightarrow B$ A $\neg A$ B $\neg B$	$\neg(A \leftrightarrow B)$ A $\neg A$ $\neg B$ B

Tabell 3: Nedbrytningsreglerna för semantiska träd

Ett semantiskt träd där en formel A förekommer överst (i "roten") kallas ett *semantiskt träd för* A. Ett semantiskt träd är alltså ett sätt att finna värderingar som satisfierar A. Om alla grenar i ett träd är slutna kallar vi trädet *slutet*. Formeln överst i trädet är då inte satisfierbar. Vi kan använda det här för att visa att en formel är en tautologi. Nämligen, om $\neg A$ inte är satisfierbar så måste A vara tautolog. Ett slutet träd för formeln $\neg A$ kallas ett semantiskt bevis för A. Semantiska bevis baserar sig på följande teorem (vars bevis finns t.ex. i [**?**])

Teorem 31. *En satslogisk formel har ett semantiskt bevis om och endast om den är tautolog.*

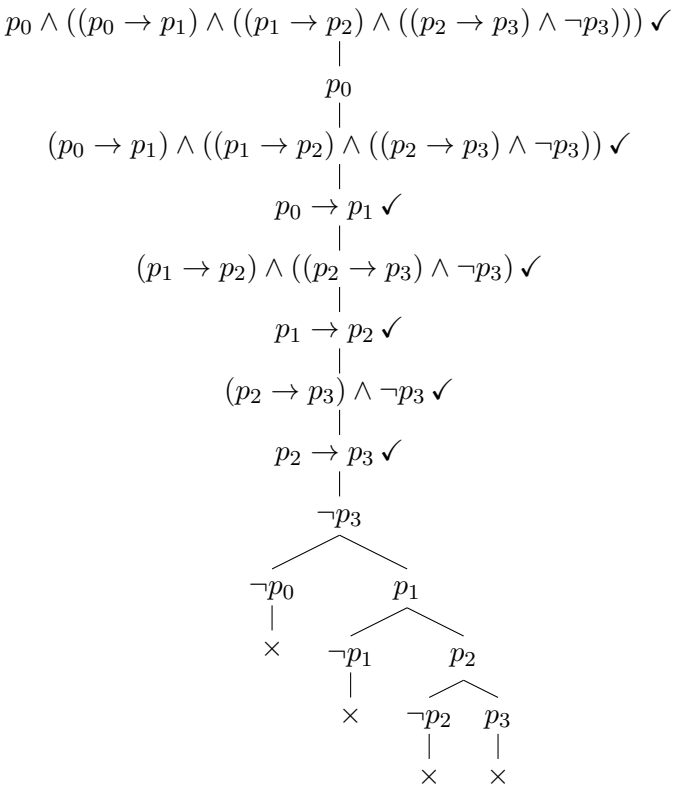
Exempel 32. Vi visar att $(A \to B) \vee (B \to A)$ är en tautologi. Vi konstruerar ett semantiskt träd för *negationen* av formeln:

$$\begin{array}{c} \neg((A \to B) \vee (B \to A))\,\checkmark \\ | \\ \neg(A \to B)\,\checkmark \\ | \\ \neg(B \to A)\,\checkmark \\ | \\ A \\ | \\ \neg B \\ | \\ B \\ | \\ \neg A \\ | \\ \times \end{array}$$

Eftersom trädet är slutet kan ingen värdering satisfiera $\neg((A \to B) \vee (B \to A))$ så $(A \to B) \vee (B \to A)$ måste vara en tautologi.

Exempel 33. Vi återvänder till exemplet i början av avsnittet och visar med ett semantiskt träd att formeln $(p_0 \wedge ((p_0 \to p_1) \wedge ((p_1 \to p_2) \wedge ((p_2 \to p_3) \wedge \neg p_3))))$ inte är satisfierbar.

Vi börjar med att bryta ner konjunktionerna en efter en. Sedan bryter vi ner implikationerna. Då vi på en gren får både en formel och dess negation kan vi sluta den (markera med $\times$) och behöver efter det inte tillämpa fler regler på den grenen. (Vi får naturligtvis tillämpa reglerna först och se efter vilka grenar som sluter sig först efteråt, men genom att sluta grenar an efter kan vi förenkla trädet avsevärt.)



Alla grenar sluter sig, så formeln är inte satisfierbar.