IIC2213 — Lógica para Ciencia de la Computación

Notas Semana 6

1. Monotonía

Recuerda que la clase pasada vimos que un conjunto Σ de formulas iba a ser insatisfacible o inconsistente si y solo si de Σ podíamos deducir cualquier cosa (incluidas las contradicciones). Esto es una consecuencia particular de este teorema más general:

Teorema (Monotonía). Sean Σ_1 y Σ_2 dos conjuntos de fórmulas en $L(P)$, y φ una fórmula en $L(P)$. Se tiene que:

$$\text{si } \Sigma_1 \models \varphi, \text{ entonces } \Sigma_1 \cup \Sigma_2 \models \varphi.$$

En palabras, el teorema dice que, si ya podíamos deducir φ de la información que teníamos en Σ_1, entonces si agregamos más información vamos a seguir pudiendo deducir φ.

Ejemplo. Para ver lo contraintuitivo que puede llegar a ser este teorema, supón nuevamente un conjunto $P = \{\texttt{está\_lloviendo}, \texttt{hace\_calor}, \texttt{hay\_smog}\}$. Imagina que sabemos que las siguientes fórmulas (agrupadas en Σ_1) son ciertas: $\Sigma_1 = \{\texttt{está\_lloviendo}, \texttt{está\_lloviendo} \to \neg\texttt{hay\_smog}\}$. La información de Σ_1 es: *está lloviendo*, y *si llueve entonces no hay smog*. Lógicamente, de Σ_1 podemos deducir que *no hay smog*, lo que formalmente se escribe como $\Sigma_1 \models \neg\texttt{hay\_smog}$. Pero si agregamos a Σ_1 la fórmula $(\texttt{hay\_smog} \wedge \texttt{hace\_calor})$, entonces el teorema de monotonía nos dice que

$$\{\texttt{está\_lloviendo}, (\texttt{está\_lloviendo} \to \neg\texttt{hay\_smog}), (\texttt{hay\_smog} \wedge \texttt{hace\_calor})\} \models \neg\texttt{hay\_smog}.$$

es decir, sabemos que hay smog y hace calor, y deducimos que no hay smog. ¡Esto no tiene sentido en el mundo real! Lo que pasa es que los humanos somos expertos en razonamiento no-monotónico. La lógica proposicional, en cambio, una vez que recibe información contradictoria, nos deja de ser útil.

Demostración de Monotonía. Sean Σ_1, Σ_2 y φ como en el enunciado del teorema, y asumamos que $\Sigma_1 \models \varphi$. Debemos mostrar que $\Sigma_1 \cup \Sigma_2 \models \varphi$.

Para mostrar que $\Sigma_1 \cup \Sigma_2 \models \varphi$, tenemos que mostrar que toda valuación τ tal que $\tau \models \Sigma_1 \cup \Sigma_2$ es tal que $\tau \models \varphi$.

Sea entonces una valuación τ arbitraria tal que $\tau \models \Sigma_1 \cup \Sigma_2$. Mostraremos que $\tau \models \varphi$.

Si $\tau \models \Sigma_1 \cup \Sigma_2$, entonces $\tau \models \psi$ para cada fórmula ψ en $\Sigma_1 \cup \Sigma_2$, y en particular para cada fórmula en Σ_1, y por tanto $\tau \models \Sigma_1$.

Hemos asumido que $\Sigma_1 \models \varphi$. Por definición, esto significa que para toda valuación η tal que $\eta \models \Sigma_1$, se tiene que $\eta \models \varphi$.

En particular, la valuación τ que elegimos verifica que $\tau \models \Sigma_1$, y por tanto $\tau \models \varphi$, que era lo que buscábamos.

Como la elección de τ es arbitraria, esto demuestra que toda valuación τ tal que $\tau \models \Sigma_1 \cup \Sigma_2$ es tal que $\tau \models \varphi$. $\square$

Monotonía y mundos posibles. Recordemos que *modelos*(φ) es una representación de todas las valuaciones τ que satisfacen a φ, a través los conjuntos S_τ. Definamos nuevamente, para cada valuación τ de P, el conjunto $S_\tau = \{p \in P \mid \tau(p) = 1\}$. Antes definimos *modelos*(φ) como:

$$\mathit{modelos}(\varphi) = \{S_\tau \mid \tau : P \to \{0,1\} \text{ y } \tau \models \varphi\}.$$

Ahora definimos *modelos*(Σ) como el conjunto de todos los S_τ tal que $\tau \models \Sigma$, es decir, como:

$$\mathit{modelos}(\Sigma) = \{S_\tau \mid \tau : P \to \{0,1\} \text{ y } \tau \models \Sigma\}.$$

De esta forma podemos redefinir la noción de consecuencia lógica:

$$\Sigma \models \varphi \text{ si y solo si } \mathit{modelos}(\Sigma) \subseteq \mathit{modelos}(\varphi).$$

Y podemos redefinir el teorema de monotonía de forma que sea más fácil entenderlo desde el punto de vista de mundos posibles: mientras más conocimiento agregamos, los mundos posibles se mantienen o desaparecen, pero nunca agregamos otro mundo posible. Así, mientras más conocimiento agreguemos, va a llegar un momento en el que no van a existir mundos posibles que estén de acuerdo con todo lo que sabemos; en ese caso hemos llegado a una contradicción.

Teorema (Monotonía 2). Sean Σ_1 y Σ_2 dos conjuntos de fórmulas en $L(P)$. Entonces *modelos*$(\Sigma_1 \cup \Sigma_2) \subseteq$ *modelos*(Σ_1).

Ejercicio. Demuestra este teorema.

2. Fórmulas, información y monotonía

Recuerda la dualidad entre un conjunto Σ de fórmulas en $L(P)$ y la noción de mundos posibles: Σ representa un conjunto S de mundos posibles.

De acuerdo al teorema de Monotonía 2, mientras más fórmulas agregamos a Σ, lo que va pasando es que la cantidad de mundos posibles disminuye. ¿Por que le llamamos entonces monotonía a este teorema? Una interpretación es que, a pesar de que los mundos posibles disminuyen, nuestra *información* siempre aumenta.

Cuando Σ solo tiene un mundo posible, podemos pensar en que representa la información perfecta: el valor de verdad de cada proposición $p \in P$ está dado. Al contrario, si Σ admite los 2^{2^n} mundos posibles, asumiendo que P tiene n proposiciones, entonces eso no es información, realmente no sabemos nada. Podemos entonces entregar una tercera modalidad de monotonía:

Teorema (Monotonía 3). Supongamos que Σ representa el conocimiento actual del mundo de un agente, en la forma de todos los mundos posibles que son consistentes con la información del agente. Cuando el agente realiza una nueva observación, sea φ la formula que representa todos los mundos posibles de acuerdo con la observación φ. Entonces $modelos(\Sigma \cup \{\varphi\}) = modelos(\Sigma) \cap modelos(\varphi)$. Además,

- Si $modelos(\Sigma) \cap modelos(\varphi) \neq \emptyset$, entonces $modelos(\Sigma \cup \varphi) \subseteq modelos(\Sigma)$: la información aumenta con la nueva observación.
- Si $modelos(\Sigma) \cap modelos(\varphi) = \emptyset$, entonces $modelos(\Sigma \cup \varphi) = \emptyset$. El sistema entra en contradicción por que la observación φ no está de acuerdo con nuestro conocimiento.

Hay una versión extendida de las notas para introducir al problema de actualización de conocimiento en forma no monótona: qué hacer cuando la nueva observación no es consistente con los mundos posibles de nuestro conocimiento. Por otro lado, en muchas aplicaciones se usan lógicas no-monótonas, cuyas reglas de conectivos hacen que el teorema de monotonía sea falso.

Para este curso, sin embargo, nos va a bastar con entender un problema fundamental: decidir si acaso $modelos(\Sigma) \cap modelos(\varphi) \neq \emptyset$

3. Resolución proposicional

Dado un conjunto Σ de fórmulas en $L(P)$ y una fórmula φ en $L(P)$, nos interesan dos problemas:

- Decidir cuando $\Sigma \models \varphi$
- Decidir cuando $modelos(\Sigma) \cap modelos(\varphi) \neq \emptyset$.

Dado un conjunto (finito) $\Sigma = \{\varphi_1, \ldots, \varphi_n\}$ de fórmulas, usamos $\bigwedge \Sigma$ para denotar la fórmula $\varphi_1 \wedge \varphi_2 \wedge \cdots \wedge \varphi_n$.

Observación. Demuestra lo siguiente:

- $\Sigma \models \varphi$ si y solo si la fórmula $(\bigwedge \Sigma) \rightarrow \varphi$ es una tautología, y por tanto $\neg\varphi \wedge (\bigwedge \Sigma)$ es una contradicción.
- $modelos(\Sigma) \cap modelos(\varphi) = \emptyset$ si y solo si $\Sigma \cup \{\varphi\}$ es una contradicción.

Entonces, parece ser que un problema interesante es ver si acaso un conjunto de fórmulas es o no una contradicción!

Conceptos preliminares

Necesitamos un poco de notación; de aquí en adelante vamos a fijar, por simpleza, un conjunto P de proposiciones. Un *literal* ℓ es una proposición $p \in P$ o la negación $\neg p$ de una proposición $p \in P$. Una *cláusula* en $L(P)$ es una disyunción de literales. Por ejemplo $(p \ \vee \ q \ \vee \ r)$ y $(p \ \vee \ \neg q \ \vee \ \neg r \vee \ q)$ son cláusulas. Una fórmula está en forma normal conjuntiva, o *Conjunctive Normal Form* (CNF) si es una conjunción de cláusulas, es decir, de la forma $C_1 \wedge C_2 \wedge \cdots \wedge C_n$, donde cada C_i es una cláusula.

Ejercicio. Muestra que toda fórmula en $L(P)$ es equivalente a una fórmula en CNF. Así mismo, muestra que todo conjunto Σ de fórmulas en $L(P)$ es equivalente a un conjunto de cláusulas.

En vista del resultado anterior, vamos a definir un sistema que sólo es capaz de verificar si $\Sigma \models \varphi$ cuando todas las fórmulas en Σ y φ están en CNF. Además, el siguiente resultado nos permite concentrarnos en un sistema capaz de verificar cuando un conjunto de cláusulas forman una contradicción (forman un conjunto inconsistente):

Ejercicio. Muestra que $\Sigma \models \varphi$ si y solo si $\Sigma \cup \{\neg\varphi\}$ es una contradicción.

Resolución

La idea de resolución es continuar generando formulas adicionales a través de un conjunto de reglas, hasta encontrar un símbolo $\perp$. Si encontramos ese símbolo, entonces sabremos que un conjunto Σ es inconsistente.

Regla de resolución. La regla de resolución es simple, dice así: Si tengo cláusulas C_1, C_2, C_3 y C_4 y una proposición $p \in P$, entonces desde $C_1 \vee p \vee C_2$ y $C_3 \vee \neg p \vee C_4$ puedo deducir $C_1 \vee C_2 \vee C_3 \vee C_4$. Escribimos eso de la siguiente forma:

$$\begin{array}{l} C_1 \vee p \vee C_2 \\ C_3 \vee \neg p \vee C_4 \\ \hline C_1 \vee C_2 \vee C_3 \vee C_4 \end{array}$$

Ejercicio. Verifica que la regla es correcta, es decir, para cualquier conjunto $\{C_1, C_2, C_3, C_4\}$ de cláusulas en P y $p \in P$ se tiene que $\{C_1 \vee p \vee C_2, C_3 \vee \neg p \vee C_4\} \models C_1 \vee C_2 \vee C_3 \vee C_4$.

Ejemplo. Para mostrar que $\{p \vee q, r \vee \neg q\} \models p \vee r$, utilizamos resolución:

$$\begin{array}{l} p \vee q \\ r \vee \neg q \\ \hline p \vee r \end{array}$$

Cuando C_1, C_2, C_3 y C_4 son vacías, usamos el símbolo $\perp$ para indicar la cláusula vacía (que diremos no es satisfacible). Es decir, se tiene que

$$\begin{array}{c} p \\ \neg p \\ \hline \perp \end{array}$$

Definición. Una *demostración por resolución* de una cláusula C desde un conjunto de cláusulas Σ es una secuencia $C_1, C_2, \ldots, C_n$ de cláusulas tales que

- Para cada $i \leq n$,

- $C_i \in \Sigma$, o bien
- existen $j, k < i$ tal que C_i es producto de aplicar resolución sobre las cláusulas C_j y C_k;

- $C = C_n$

Escribimos $\Sigma \vdash C$ cuando existe una demostración por resolución de una cláusula C desde un conjunto de cláusulas Σ. Por ejemplo, ya vimos que $\{p \vee q, r \vee \neg q\} \vdash p \vee r$

Ejercicio. Sea $P = \{p, q, r, s, t\}$. Construye un conjunto Σ y una cláusula C tal que $\Sigma \vdash C$. Trata de que la secuencia de cláusulas intermedias sea lo más larga posible.

Garantías teóricas

Nada de lo que hemos definido tendría sentido si nuestro sistema de resolución no nos sirviera para detectar inconsistencias reales. Esas propiedades son formalizadas en los siguientes dos resultados, que demostraremos la próxima clase:

Teorema (correctitud de resolución). Si Σ es un conjunto de cláusulas en $L(P)$, tal que $\Sigma \vdash \bot$, entonces Σ es inconsistente.

Teorema (completitud de resolución). Si Σ es un conjunto de cláusulas en $L(P)$, tal que Σ es inconsistente, entonces $\Sigma \vdash \bot$.

El primer teorema nos dice que resolución es un algoritmo *correcto*: si detectamos una demostración por resolución que parte en Σ y llega a deducir $\bot$, entonces Σ es efectivamente inconsistente. El segundo teorema nos dice que el algoritmo es *completo*: si un conjunto es inconsistente entonces sabemos que podremos demostrarlo por resolución. Juntando los dos teoremas se tiene que Σ es inconsistente si y solo si $\Sigma \vdash \bot$.

Extensiones

Muchas veces resulta incómodo trabajar solo con la regla de resolución, por lo que uno le añade más reglas al sistema deductivo. En nuestro caso podemos, por ejemplo, añadir la regla de factorización:

$$\frac{C_1 \vee p \vee C_2 \vee p \vee C_3}{C_1 \vee p \vee C_2 \vee C_3}$$

Regla de factorizacion. Si tengo una cláusula C y una proposición $p \in P$, entonces desde $p \vee p \vee C$ puedo deducir $C \vee p$. Con eso podemos armar otro sistema deductivo:

Definición. Una *demostración por resolución y factorización* de una cláusula C desde un conjunto de cláusulas Σ es una secuencia $C_1, C_2, \ldots, C_n$ de cláusulas tales que

- Para cada $i \leq n$,
  - $C_i \in \Sigma$, o bien

- existen $j, k < i$ tal que C_i es producto de aplicar resolución sobre las cláusulas C_j y C_k, o bien
- existe $k < i$ tal que C_i es producto de aplicar factorización sobre C_k;

- $C = C_n$

Abusando notación, también escribimos $\Sigma \vdash C$ cuando existe una demostración por resolución y factorización de una cláusula C desde un conjunto de cláusulas Σ. Por ejemplo, ya vimos que $\{p \vee q, r \vee \neg q\} \vdash p \vee r$

Ejercicio. ¿Puedes pensar en otras reglas que podrían ser interesantes de tener en resolución? Crea al menos una regla, defínela formalmente, y re-define tu sistema de resolución usando además esa regla. OJO: Debes asegurarte que la regla es correcta, por que si no tu sistema no va a ser correcto.