UNIVERSIDAD DE CHILE
ESCUELA DE INGENIERIA
DEPARTAMENTO DE CIENCIAS DE LA COMPUTACION

Inteligencia Artificial - CC52A
Guía 1

1. Dado $\alpha, \beta \in L(P)$, demuestre que $\alpha \models \beta$ y $\beta \models \alpha$ si y sólo si $\models (\alpha \leftrightarrow \beta)$ (es decir, $\alpha \models \beta$ y $\beta \models \alpha$ si y sólo si $(\alpha \leftrightarrow \beta)$ es una tautología).

2. Dado $\Sigma \subseteq L(P)$ y $\alpha, \beta \in L(P)$, demuestre que si α es una tautología, entonces $\Sigma \cup \{\alpha\} \models \beta$ si y sólo si $\Sigma \models \beta$.

3. Dado $\Sigma \subseteq L(P)$ y $\varphi, \psi, \theta \in L(P)$, demuestre que si $\varphi \to \psi$ es una tautología, entonces $\Sigma \cup \{\varphi, \psi\} \models \theta$ si y sólo si $\Sigma \cup \{\varphi\} \models \theta$.

4. Dado $\Sigma \subseteq L(P)$ y $\alpha, \beta \in L(P)$ tal que α y $\Sigma \cup \{\beta\}$ no tienen variables proposicionales en común. ¿Es cierto que $\Sigma \models \beta$ si y sólo si $\Sigma \cup \{\alpha\} \models \beta$?

5. Sea p una proposición atómica en P. Para cada $\alpha, \beta \in L(P)$ defina recursivamente la *fórmula obtenida desde α reemplazando p por β*, denotado por $\alpha[\beta/p]$, como sigue:

   - $\alpha[\beta/p] \;=\; \begin{cases} \alpha & \text{si } \alpha \text{ es proposición atómica y } \alpha \neq p \\ \beta & \text{si } \alpha = p \end{cases}$
   - $(\alpha_1 \vee \alpha_2)[\beta/p] \;=\; \alpha_1[\beta/p] \vee \alpha_2[\beta/p]$
   - $(\neg\alpha)[\beta/p] \;=\; \neg\alpha[\beta/p]$

   Demuestre que si α_1 y α_2 son equivalentes entonces para todo $\beta \in L(P)$ se tiene que $\beta[\alpha_1/p]$ y $\beta[\alpha_2/p]$ también son equivalentes.

6. Defina recursivamente el *dual* de una fórmula ϕ, denotado por ϕ^*, como sigue:

   - $p^* = \neg p$, si $p \in P$;
   - $(\alpha \vee \beta)^* = \alpha^* \wedge \beta^*$;
   - $(\alpha \wedge \beta)^* = \alpha^* \vee \beta^*$;
   - $(\neg\alpha)^* = \neg\alpha^*$.

   Demuestre que para todo ϕ en $L(P)$ se tiene que ϕ^* es equivalente a $\neg\phi$.

7. Sea p una proposición atómica en P. Para cada $\alpha, \beta \in L(P)$ defina recursivamente la *fórmula obtenida desde α reemplazando p por β*, denotado por $\alpha[\beta/p]$, como sigue:

   - $\alpha[\beta/p] \;=\; \begin{cases} \alpha & \text{si } \alpha \text{ es proposición atómica y } \alpha \neq p \\ \beta & \text{si } \alpha = p \end{cases}$
   - $(\alpha_1 \vee \alpha_2)[\beta/p] \;=\; \alpha_1[\beta/p] \vee \alpha_2[\beta/p]$

- $(\neg\alpha)[\beta/p] \ = \ \neg\alpha[\beta/p]$

Demuestre que si α_1 y α_2 son equivalentes entonces para todo $\beta \in L(P)$ se tiene que $\beta[\alpha_1/p]$ y $\beta[\alpha_2/p]$ también son equivalentes.

8. Defina recursivamente el *dual* de una fórmula ϕ, denotado por ϕ^*, como sigue:

   - $p^* = \neg p$, si $p \in P$;
   - $(\alpha \vee \beta)^* = \alpha^* \wedge \beta^*$;
   - $(\alpha \wedge \beta)^* = \alpha^* \vee \beta^*$;
   - $(\neg\alpha)^* = \neg\alpha^*$.

   Demuestre que para todo ϕ en $L(P)$ se tiene que ϕ^* es equivalente a $\neg\phi$.

9. Sea $\oplus$ el conectivo lógico binario definido como sigue: Para una valuación $\sigma : P \to \{0, 1\}$ se tiene que $\sigma(\alpha \oplus \beta) = 1$ si y sólo si $\sigma(\alpha) = 1 - \sigma(\beta)$.

   ¿Es el conjunto $\{\oplus, \leftrightarrow\}$ de conectivos lógicos funcionalmente completo?

10. El conectivo lógico `NOR` es definido de la siguiente forma:

p	q	p `NOR` q
0	0	1
0	1	0
1	0	0
1	1	0

   Demuestre que `NOR` es funcionalmente completo.

11. El conectivo ternario `MAYORIA` es definido de la siguiente forma:

p	q	r	`MAYORIA`(p, q, r)
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	1
1	1	1	1

   Demuestre que `MAYORIA` no es funcionalmente completo.

12. ¿Es $\{\neg,$ `MAYORIA`$\}$ funcionalmente completo?

13. El conectivo unario $\perp$ es definido de la siguiente forma:

p	$\perp p$
0	0
1	0

Este conectivo usualmente se denota sin la letra proposicional porque su valor de verdad es siempre 0 (por ejemplo, denotamos $p \wedge (\bot q)$ como $p \wedge \bot$).

Demuestre que $\{\neg, \texttt{MAYORIA}, \bot\}$ es funcionalmente completo.

14. Formalice el siguiente argumento en el cálculo proposicional:

    > "Si Superman fuera capaz y deseara prevenir el mal, entonces lo haría. Si Superman fuera incapaz de prevenir el mal, entonces sería impotente, y si no deseara prevenir el mal, entonces sería malévolo. Si Superman existe, no es ni impotente ni malévolo. Superman no previene el mal. Entonces, Superman no existe."

    Demuestre usando resolución que Superman no existe.

15. Decimos que una fórmula φ está en 3-CNF si φ está en CNF y cada una de sus claúsulas contiene a lo más tres literales. Por ejemplo, $(p \vee q \vee \neg r) \wedge (\neg p \vee s)$ está en 3-CNF mientras que $(p \vee \neg q \vee \neg r \vee s)$ no está en 3-CNF.

    Demuestre que existen fórmulas que no son equivalentes a ninguna fórmula en 3-CNF.

16. Decimos que una fórmula φ está en k-CNF ($k \geq 2$) si φ está en CNF y cada una de sus claúsulas contiene a lo más k literales. ¿Existe algún valor de k para el cual toda fórmula es equivalente a una fórmula en k-CNF?

17. Considere una valuación $\sigma : P \rightarrow \{0, 1\}$. Demuestre que el conjunto $S = \{\phi \mid \phi \in L(P),\ \sigma(\phi) = 1\}$ satisface lo siguiente:

    Para todo conjunto S' de fórmulas en $L(P)$ se tiene que si $S \subseteq S'$ y S' es satisfacible, entonces $S = S'$.

18. Estudie detenidamente la demostración de la completidad débil del método de resolución para lógica proposicional.

19. Sea C un conjunto de cláusulas proposicionales y m un literal, i.e. una variable proposicional o la negación de una variable proposicional.

    Decimos que el conjunto de clásulas C deriva la cláusula c *usando k pasos de resolución proposicional*, $k \geq 0$, si existe secuencia $c_0, c_1, \ldots, c_k = c$ de cláusulas tal que para cada $0 \leq i \leq k$ se tiene lo siguiente:

    (a) $c_i \in C$; ó
    (b) existen $j, k < i$ tal que c_i se obtiene mediante resolución proposicional desde c_j y c_k.

    Defina $C(m)$ como el siguiente conjunto de cláusulas:

    $$C(m) \ = \ \{c \mid c \in C, m \notin c, \bar{m} \notin c\} \cup \{(c - \bar{m}) \mid c \in C, m \notin c, \bar{m} \in c\}.$$

    Recuerde que para literal m, $\bar{m} = \neg p$ si $m = p$ y $\bar{m} = p$ si $m = \neg p$.

    Demuestre lo siguiente:

a) Si se puede derivar la cláusula c desde $C(m)$ usando k pasos de resolución proposicional, $k \geq 0$, entonces se puede derivar la cláusula c' desde C usando k pasos de resolución proposicional, donde c' es la cláusula c ó la cláusula $c \cup \{\bar{m}\}$.

b) Si $C(m)$ deriva la cláusula vacía $\square$ usando k_1 pasos de resolución proposicional, $k_1 \geq 0$, y $C(\bar{m})$ deriva la cláusula vacía $\square$ usando k_2 pasos de resolución proposicional, $k_2 \geq 0$, entonces C deriva la cláusula vacía $\square$ usando a lo más $k_1 + k_2 + 1$ pasos de resolución proposicional.

En los siguientes ejercios decimos que un algoritmo es *eficiente*, si el número de pasos ejecutado por el algoritmo es n^c cuando la entrada tiene largo n, donde c es una constante. Por ejemplo, un algoritmo que funciona en tiempo n^2 es eficiente mientras que un algoritmo que funciona en tiempo 2^n no lo es.

1. Encuentre un algoritmo eficiente que dada una fórmula φ en CNF construye una fórmula ψ en 3-CNF tal que φ es satisfacible si y sólo si ψ es satisfacible.

2. Encuentre un algoritmo eficiente que verifique si una fórmula en DNF es satisfacible.

3. Encuentre un algoritmo eficiente que verifique si una fórmula en CNF es una tautología.

4. Un grafo G es una tupla (N, A), donde N es un conjunto de nodos y $A \subseteq N \times N$ es un conjunto de arcos. Un grafo es no dirigido si cada vez que $(a, b) \in A$ se tiene que $(b, a) \in A$.

   Un grafo no dirigido $G = (N, A)$ es 3-coloreable si existe una asignación de colores para los nodos tal que nodos adyacentes reciben colores distintos. Formalmente, G es 3-coloreable si existe una función $f : N \to \{\mathit{blanco}, \mathit{azul}, \mathit{rojo}\}$ tal que para cada $(a, b) \in A$ se tiene que $f(a) \neq f(b)$.

   Demuestre que el problema de 3-coloración puede ser reducido eficientemente al problema de satisfacibilidad. Vale decir, encuentre un algoritmo eficiente que dado un grafo G construye una fórmula φ tal que G es 3-coloreable si y sólo si φ es satisfacible. Estime el número de pasos de su algoritmo cuando el grafo G tiene n nodos y m arcos.

5. Un grafo no dirigido $G = (N, A)$ es k-coloreable ($k \geq 2$) si existe una función $f : N \to \{1, 2, \ldots, k\}$ tal que para cada $(a, b) \in A$ se tiene que $f(a) \neq f(b)$.

   Demuestre que el problema de k-coloración puede ser reducido eficientemente al problema de satisfacibilidad. Estime el número de pasos de su algoritmo para un grafo con n nodos y m arcos.

6. Suponga que existe un algoritmo eficiente para el problema de satisfacibilidad, es decir, un algoritmo que dada una fórmula φ con n letras proposicionales, en n^c pasos retorna 1 si la fórmula es satisfacible y 0 en caso contrario, donde c es una constante fija.

   Encuentre un algoritmo eficiente que dada una fórmula φ retorna una valuación σ tal que $\sigma(\varphi) = 1$ si φ es satisfacible, y retorna 0 en caso contrario.

7. Decimos que dos grafos $G_1 = (N_1, A_1)$ y $G_2 = (N_2, A_2)$ son *isomorfos* si existe una biyección $f : N_1 \to N_2$ tal que para todo a y b en A_1 se tiene que $(a, b) \in A_1$ si y sólo si $(f(a), f(b)) \in A_2$.

   Encuentre un algoritmo eficiente que dados dos grafos G_1 y G_2 construye una fórmula φ tal que G_1 y G_2 son isomorfos si y sólo si φ es satisfacible. Estime el número de pasos de su algoritmo cuando G_1 tiene n_1 nodos y m_1 arcos, y G_2 tiene n_2 nodos y m_2 arcos.

8. Un *clique* es un grafo G tal que para todo par de nodos u, v en G se tiene que existe un eje entre ellos.

   Demuestre que para cada fórmula ϕ en CNF uno puede encontrar eficientemente un grafo G y un entero k, tal que ϕ es satisfacible si y sólo si G contiene un clique de tamaño k.

9. Dada una matriz C de 3×3 que contiene números entre 0 y 3, decimos que C es *completable* si es que existe una manera de reemplazar los números 0 por números entre 1 y 3 de tal forma que la suma de cada fila y de cada columna es la misma. Por ejemplo, la siguiente matriz es completable:

2	0	0
0	2	0
0	0	3

   puesto que podemos reemplazar los valores 0 por los siguientes valores:

2	2	1
2	2	1
1	1	3

   de manera tal que la suma de cada fila y de cada columna es 5. En cambio, la siguiente matriz no es completable:

1	1	1
0	0	0
3	0	0

   Dada una matriz C de 3×3, construya una fórmula φ en lógica proposicional tal que C es completable si y sólo si φ es satisfacible. En particular, φ tiene que ser construida de tal forma que cada valuación σ que satisface a φ represente una forma de completar C.

10. El *principio de los cajones* establece que si $n+1$ objetos son distribuidos en n cajones, entonces al menos habrá un cajón con más de un objeto.

    Demuestre el principio para $n = 2$ usando cálculo proposicional y resolución.

Los siguientes ejercicios tratan acerca de la lógica de primer orden:

1. Sea $\mathcal{L} = \{E(\cdot, \cdot)\}$ un lenguaje utilizado para representar grafos. En cada una de las siguientes preguntas escriba una $\mathcal{L}$-oración que represente la propiedad mencionada. En otras palabras, escriba una $\mathcal{L}$-oración ϕ tal que una $\mathcal{L}$-estructura $\mathcal{A}$ satisface ϕ si y sólo si $\mathcal{A}$ tiene la propiedad mencionada.

   (a) El grafo es un clique.

   (b) El grafo contiene un clique con 4 nodos.

   (c) El grafo tiene un ciclo con 4 nodos.

   (d) Existen elementos en el grafo cuya distancia es 4.

(e) La distancia máxima entre dos nodos del grafo es 3.

2. Sea $\mathcal{L} = \{f(\cdot), c\}$, donde f es una función unaria, y sean φ_1, φ_2, φ_3 las siguientes $\mathcal{L}$-oraciones:

$$\begin{aligned}\varphi_1 &= \forall x \forall y\ (f(x) = f(y) \to x = y),\\ \varphi_2 &= \forall x (f(x) \neq c),\\ \varphi_3 &= \forall x (x \neq c \to \exists y\ f(y) = x).\end{aligned}$$

(a) Muestre una $\mathcal{L}$-estructura $\mathcal{A}_1$ tal que $\mathcal{A}_1 \models \varphi_1 \wedge \varphi_2 \wedge \varphi_3$.
(b) Muestre una $\mathcal{L}$-estructura $\mathcal{A}_2$ tal que $\mathcal{A}_2 \models \neg\varphi_1 \wedge \varphi_2 \wedge \varphi_3$.
(c) Muestre una $\mathcal{L}$-estructura $\mathcal{A}_3$ tal que $\mathcal{A}_3 \models \varphi_1 \wedge \neg\varphi_2 \wedge \varphi_3$.
(d) Muestre una $\mathcal{L}$-estructura $\mathcal{A}_4$ tal que $\mathcal{A}_4 \models \varphi_1 \wedge \varphi_2 \wedge \neg\varphi_3$.

3. Al igual que en la lógica proposicional, decimos que dos oraciones φ y ψ son *equivalentes*, denotado como $\varphi \equiv \psi$, si para toda estructura $\mathcal{A}$ se tiene que: $\mathcal{A} \models \varphi$ si y sólo si $\mathcal{A} \models \psi$.

¿Cuáles de las siguientes afirmaciones son válidas? Justifique su respuesta.

(a) $\forall x\, \varphi \ \equiv\ \neg\exists x \neg\varphi$.
(b) $\exists x\, \varphi \ \equiv\ \neg\forall x \neg\varphi$.
(c) $\exists x\, (\varphi \vee \psi) \ \equiv\ (\exists x\, \varphi) \vee (\exists x\, \psi)$.
(d) $\forall x\, (\varphi \vee \psi) \ \equiv\ (\forall x\, \varphi) \vee (\forall x\, \psi)$.
(e) $\exists x\, (\varphi \wedge \psi) \ \equiv\ (\exists x\, \varphi) \wedge (\exists x\, \psi)$.
(f) $\forall x\, (\varphi \wedge \psi) \ \equiv\ (\forall x\, \varphi) \wedge (\forall x\, \psi)$.

4. Al igual que en la lógica proposicional, decimos que una oración φ es *consecuencia lógica* de un conjunto de oraciones Σ, denotado como $\Sigma \models \varphi$, si para cada estructura $\mathcal{A}$ tal que $\mathcal{A} \models \Sigma$, se tiene que $\mathcal{A} \models \varphi$.

¿Cuáles de las siguientes afirmaciones son válidas? Justifique su respuesta.

(a) $\{\forall x \exists y\, R(x, y)\} \ \models\ \exists x \forall y\, R(x, y)$.
(b) $\{\exists x \forall y\, R(x, y)\} \ \models\ \forall x \exists y\, R(x, y)$.
(c) $\{\forall x \forall y\, (R(x, y) \to R(y, x)),\ \forall x \forall y \forall z\, ((R(x, y) \wedge R(y, z)) \to R(x, z))\} \ \models\ \forall x\, R(x, x)$.
(d) $\{\exists x\, (P(x) \wedge Q(x))\} \ \models\ (\exists x\, P(x)) \wedge (\exists x\, Q(x))$.
(e) $\{\exists x\, P(x),\ \exists y\, Q(y))\} \ \models\ \exists x\, (P(x) \wedge Q(x))$.
(f) $\{\forall x \exists y\, S(x, y)\} \ \models\ \exists y\, S(y, y)$.

5. Sean α y β dos $\mathcal{L}$-fórmulas. Demuestre que

$$\{\ \forall x(\alpha \to \beta),\ \forall x \alpha\ \} \models \forall x \beta.$$

6. Demuestre que ninguna de las oraciones siguientes es consecuencia lógica de las dos restantes:

- $\forall x \forall y \forall z (P(x, y) \wedge P(y, z) \to P(x, z))$;
- $\forall x \forall y (P(x, y) \wedge P(y, x) \to x = y)$;

- $\forall x \forall y (P(a, x) \to P(y, b))$.

7. Sea $\mathcal{L}$ un vocabulario que consiste de una función binaria f y una constante e.

   a. Construya una $\mathcal{L}$-oración ϕ tal que $\mathcal{A} \models \phi$ si y sólo si $\mathcal{A}$ es un grupo, para toda $\mathcal{L}$-estructura $\mathcal{A}$.
   b. Demuestre que $\phi \models \forall x \forall y \exists z (f(x, z) = y)$.
   c. Demuestre que $\phi \not\models \forall x \forall y (f(x, y) = f(y, x))$.

8. Construya una oración ϕ_n tal que $\mathcal{A} \models \phi_n$ si y sólo si el dominio de $\mathcal{A}$ tiene al menos n elementos, $n \geq 1$.

9. Construya una oración ψ_n tal que $\mathcal{A} \models \psi_n$ si y sólo si el dominio de $\mathcal{A}$ tiene exactamente n elementos, $n \geq 1$.

10. Sea $\mathcal{L}$ un vocabulario que consiste de una función binaria f y una constante e. Sea ϕ la oración tal que $\mathcal{A} \models \phi$ si y sólo si $\mathcal{A}$ es un grupo, para toda $\mathcal{L}$-estructura $\mathcal{A}$.

    a) Demuestre <u>usando resolución</u> que $\forall x \forall y \exists z (f(x, z) = y)$ es consecuencia lógica de ϕ.
    b) Demuestre <u>usando resolución</u> que $\forall x \forall y \exists z \forall z (f(x, y) = f(x, z) \to y = z)$ es consecuencia lógica de ϕ.

    Es necesario primero skolemizar las oraciones, luego pasarlas a forma de cláusula, y por último integrar los axiomas para la igualdad.

11. Sea $\mathcal{L}$ un vocabulario que consiste de una relación binaria R.

    a) Construya una oración ϕ que se satisface sólo en aquellas $\mathcal{L}$-estruturas donde la relación R es total (es decir, para cada elemento a existe un elemento b tal que $R(a, b)$), simétrica, y transitiva.
    b) Demuestre usando resolución que $\forall x R(x, x)$ es consecuencia lógica de ϕ.

12. Demuestre que al algoritmo visto en clases para computar el umg de un conjunto de expresiones es correcto y completo, i.e. que su salida es un unificador cuando éste existe, y que además en ese caso el unificador computado es un umg.

13. Para los siguientes conjuntos de expresiones, determine si existe un umg o no, y encuéntrelo cuando existe.

    a) $\{P(f(y), w, g(z)),\ P(u, u, v)\}$;
    b) $\{P(f(y), w, g(z)),\ P(v, u, v)\}$;
    c) $\{P(a, x, f(g(y))),\ P(z, h(z, w), f(w))\}$;

14. Una sustitución θ es *idempotente* si $\theta = \theta\theta$.

    Sea $\theta = \{x_1/t_1, \ldots, x_n/t_n\}$, y asuma que V es el conjunto de variables mencionadas en $\{t_1, \ldots, t_n\}$. Demuestre que θ es idempotente si y sólo si $\{x_1, \ldots, x_n\} \cap V = \emptyset$.

15. Demuestre que todo umg producido por el algoritmo visto en clases es idempotente.

16. Asuma que θ_1 y θ_2 son sustituciones, y que existen sustituciones σ_1 y σ_2 tales que $\theta_1 = \theta_2\sigma_1$ y $\theta_2 = \theta_1\sigma_2$.

    Demuestre que existe una sustitución γ que es variable pura (es decir, es de la forma $\{x_1/t_1, \ldots, x_n/t_n\}$, donde cada t_i, $1 \leq i \leq n$, es una variable), tal que $\theta_1 = \theta_2\gamma$.

17. Sea Σ un conjunto de oraciones. Demuestre que existe un conjunto Σ' de cláusulas tal que Σ es satisfacible si y sólo si Σ' es satisfacible.

18. Encuentre un conjunto de cláusulas Σ, tal que la intersección de todos sus modelos de Herbrand no es un modelo de Σ.

19. En este ejercicio puede asumir la disponibilidad de un predicado $\text{Menor}(x, y)$ que significa que el elemento x es menor o igual que y.

    - Defina el predicado $\text{INSERT}(x, l, l')$ en PROLOG, donde l es una lista ordenada en orden no decreciente, y l' es el resultado de insertar el elemento x en la lista l, preservando el orden no decreciente.
    - Defina un predicado $\text{MERGE}(l_1, l_2, l)$ en PROLOG, donde l_1 y l_2 son listas ordenadas, y l es el resultado de mezclar l_1 y l_2.

20. Recuerde que la clase de las expresiones regulares sobre alfabeto finito Σ se define recursivamente como sigue:

    a) $\emptyset$ y ε son expresiones regulares sobre Σ;
    b) para cada $a \in \Sigma$ se tiene que a es una expresión regular sobre Σ;
    c) si e_1 y e_2 son expresiones regulares sobre Σ, entonces $(e_1 \cup e_2)$ y $(e_1 \cdot e_2)$ son expresiones regulares sobre Σ;
    d) si e es una expresión regular sobre Σ, entonces e^* es una expresión regular sobre Σ.

    Por tanto, podemos pensar cada expresión regular sobre alfabeto Σ como un término sin variables libres sobre el vocabulario que contiene a las constantes $\emptyset$, ε, y $\{a \mid a \in \Sigma\}$, a las funciones binarias $(\cup)$ y $(\cdot)$, y a la función unaria $()^*$.

    Con cada expresión regular e sobre Σ asociamos un subconjunto $L(e)$ de Σ^* (el *lenguaje regular asociado con la expresión* e). Este se define recursivamente de la siguiente forma:

    a) $L(\emptyset) = \emptyset$, y $L(\varepsilon) = \varepsilon$, asumiendo que ε es la palabra vacía;
    b) $L(a) = \{a\}$;
    c) $L(e_1 \cup e_2) = L(e_1) \cup L(e_2)$, y $L(e_1 \cdot e_2) = \{w \in \Sigma^* \mid w = w_1w_2,\ w_1 \in L(e_1),\ w_2 \in L(e_2)\}$, asumiendo que w_1w_2 es la concatenación de las palabras w_1 y w_2;
    d) $L(e^*) = \{\varepsilon\} \cup \{w \in \Sigma^* \mid \text{para algún } n \geq 1,\ w = w_1w_2 \ldots w_n \text{ y para cada } 1 \leq i \leq n,\ w_i \in L(e)\}$.

    Para cada uno de los siguientes predicados construya un programa en PROLOG que lo defina:

    (a). El predicado *primero*(x, y), que se satisface si y sólo si el símbolo x pertenece al siguiente conjunto: $\{a \in \Sigma \mid \text{ existe } w \in \Sigma^* \text{ tal que } aw \in L(y)\}$.

(b). El predicado *ultimo*(x, y), que se satisface si y sólo si el símbolo x pertenece al siguiente conjunto: $\{a \in \Sigma \mid \text{ existe } w \in \Sigma^* \text{ tal que } wa \in L(y)\}$.

(c). El predicado *siguiente*(x, y, z), que se satisface si y sólo si el par de símbolos (x, y) pertenece al siguiente conjunto: $\{(a, b) \in \Sigma \times \Sigma \mid \text{ existen } w_1, w_2 \in \Sigma^* \text{ tal que } w_1 ab w_2 \in L(z)\}$.

Para esto puede asumir que tiene disponible un predicado $V(x)$ que se satisface si y sólo si $\varepsilon \in L(x)$.

21. Estudie PROLOG con detención, tanto los problemas presentados en clase y ayudantía como otros que pueda encontrar en la web (la página

`http://wiki.python.org/moin/ProblemSets/99_Prolog_Problems_Solutions`

es particularmente interesante).