



Facultad de Ingeniería  
Universidad Nacional de la Patagonia San Juan Bosco



## **TALLER DE NUEVAS TECNOLOGIAS**

### **TRABAJO FINAL**

#### **DINÁMICA DE CONOCIMIENTO IMPLEMENTACIÓN DE UN SISTEMA DE CAMBIO**

##### **Integrantes:**

- APU Cobo Medvedsky, Elias Daniel
- APU Ligorria, Alexis
- APU Suazo, Leonardo Ezequiel

##### **Profesor:**

- Dr. Falappa, Marcelo A.

**CICLO LECTIVO 2015**

## TABLA DE CONTENIDO

Enunciado .....	4
Ejercicios y Resoluciones .....	5
Consideraciones .....	5
Abreviaturas.....	5
Predicados a utilizar .....	5
Predicados dinámicos .....	5
Ejercicios propuestos .....	6
Expandir(A) .....	6
Contraer(A) .....	8
Revisar(A) .....	10
Consolidar .....	11
Mostrar .....	12
TUTORIAL .....	13
Home .....	13
Aplicación.....	13
Código Fuente.....	16
EJEMPLOS.....	17
Ejemplo 1 .....	17
Expandir BC.....	17
Imprimir BC.....	17
Contraer por menos importante.....	18
Imprimir BC.....	18
Expandir e imprimir BC .....	18
Consolidar por menos importante .....	18
Imprimir BC.....	19
Ejemplo 2 .....	19
Expandir e imprimir BC .....	19
Revisar por mínima cantidad .....	19
Contraer por mantener mas importantes .....	20
Imprimir BC.....	20
Ejemplo 3 .....	20
Expandir e imprimir BC .....	21

Contraer por menos importante.....	21
Imprimir BC.....	21
Revisar por menos importante.....	21
Imprimir BC.....	22
Expandir e imprimir BC (inconsistencia) .....	22
Consolidar por mínima cantidad.....	22
Bibliografía.....	23

## ENUNCIADO

Se debe implementar en **PROLOG** un sistema de revisión de creencias que realice las siguientes operaciones:

1. Expansión: agregar una creencia  $\alpha$  a una base de conocimiento  $K$ .
2. Contracción: contraer una base de conocimiento  $K$  con respecto a  $\alpha$ .
3. Revisión Priorizada: revisar una base de conocimiento  $K$  con respecto a  $\alpha$  asegurando la propiedad de éxito.
4. Consolidación: restaurar consistencia en una base de conocimiento  $K$ .

El lenguaje debe ser un lenguaje proposicional con letras proposicionales  $a, b, \dots, z$ . El mismo debe contener los conectivos tradicionales que los notaremos como sigue:

1. Conjunción: -y-
2. Disjunción: -o-
3. Implicación: ->
4. Negación: no

El sistema debe admitir una relación de orden entre las sentencias " $\leq$ ". " $a \leq b$ " se interpretará como que " $a$  es a lo sumo tan importante como  $b$ ". Usando notación prefija,  $a \leq b$  se denotará como **ee(a,b)**.

Asumimos que se adopta el modelo **kernel contraction / revisión**, el sistema deberá mostrar ante cada cambio (salvo las expansiones) las siguientes opciones:

1. Descartar las creencias menos importantes y preservar el resto.
2. Preservar las creencias más importantes y descartar el resto.
3. Mostrar el cambio que elimine la menor cantidad de creencias.

Asumiremos que la base de conocimiento se representa mediante un predicado **kb**.

## EJERCICIOS Y RESOLUCIONES

### CONSIDERACIONES

---

#### ABREVIATURAS

- BC: base de conocimiento

---

#### PREDICADOS A UTILIZAR

- kb(a): Se utiliza para representar la BC
- ee(b,c): Se utiliza para determinar una relación de orden entre las sentencias.
  - Lógica:  $b \leq c$
  - Interpretación: “*b es a lo sumo tan importante como c*”.

---

#### PREDICADOS DINÁMICOS

- Se definen los siguientes predicados dinámicos, que permiten la utilización de los mismos en tiempo de ejecución:

`:- dynamic`

`Kb/1`

`:- dynamic`

`ee/2`

- Limpieza de la BC: Se define el siguiente predicado para limpiar la BC.

`limpiarBC :- retractall (kb(_)),  
                  retractall (ee(_,_)).`

El predicado retract consistente en:

- retract/1: elimina únicamente la primera cláusula que unifique con el argumento (siempre se elimina por el principio).
  - retractall/1 Elimina todas las cláusulas que unifiquen con el argumento.
- 
- FNC: una sentencia está en Forma Normal Conjuntiva si corresponde a un conjunto de cláusulas, donde una de estas es una disjunción de literales.
  - Deducción: Para poder contraer y revisar la BC  $K$  por  $\alpha$  hace falta conocer los subconjuntos minimales de  $K$  que deducen  $X$  ( $\alpha$ ).
    - Genera todos los posibles subconjuntos minimales de la BC.
    - Se unifica la BC con alguno de los subconjuntos de  $L$ .
    - Verificar si el subconjunto  $K_0$  “deduce”  $\alpha$ .
    - Verificar si el subconjunto  $K_0$  es mínimo.

```
deduce (X,K) :- posiblesSubconjuntos (L),  
                unificarSubconjunto (L,K),  
                deduceEn (X,K),  
                subconjuntoMinimo (X,K).
```

## EJERCICIOS PROPUESTOS

---

### EXPANDIR(A)

Debe agregar A a la BC. Cuando se expande, se debe insertar la nueva creencia y su relación de importancia.

El predicado utilizado es `expandirBC(A,[B,C])`. Donde A es la nueva creencia, B es una lista que contiene las creencias a lo sumo tan importantes como A y C es otra lista que contiene las creencias más importantes que A.

---

## DETALLE DE IMPLEMENTACIÓN

Se utiliza el predicado `asserta/1` que se emplea para insertar una nueva clausula al principio de la BC.

- Insertar una relación de importancia en particular:

```
insertarEE (A,B) :-    ee(A,B),
                      !.
insertarEE (A,B) :-    asserta(ee(A,B)).
```

- Insertar las relaciones de importancia, que contiene las creencias más importantes que A.

```
insertarMaEE ([],_) :-    !.
insertarMaEE ([H|T],A) :-    insertarMaEE (T,A), insertarEE (H,A),
                              !.
insertarMaEE (H,A) :-    insertarEE (H,A),
                              !.
```

- Inserta las relaciones de importancia, que contiene las creencias a lo sumo tan importantes como A.

```
insertarSuEE ([],_) :-    !.
insertarSuEE ([H|T],A) :-    insertarSuEE (T,A),
                              insertarEE (A,H),
                              !.
insertarSuEE (H,A) :-    insertarEE (A,H),
                              !.
```

- Insertar A en la BC.

```
expandirBC (A) :-    kb (A),
                    !.
expandirBC (A) :-    asserta (kb(A)).
```

- Expandir la BC, insertando A y las relaciones de importancia asociadas.  
 $\text{expandirBC}(A, [B, C]) \text{ :- expandirBC}(A),$   
 $\text{insertarSuEE}(C, A),$   
 $\text{insertarMaEE}(B, A).$

---

## CONTRAER(A)

Se elimina conocimiento de la base. Este puede ser implícito o explícito.

Utilizamos el modelo **kernel contraction**. Las operaciones de kernel contraction de un conjunto K con respecto a una sentencia  $\alpha$  se definen (informalmente) como la diferencia entre el conjunto original y el conjunto de elementos seleccionados por la función de incisión. Esta función de incisión “corta” cada conjunto minimal (kernel) que implica a la sentencia  $\alpha$ .

Detalle de implementación:

1. Se identifican todos los subconjuntos de K,  $K_n$  que deducen  $\alpha$ .
2. Se elimina al menos un elemento de cada subconjunto  $K_n$  dependiendo el criterio de selección elegido.

---

## DETALLE DE IMPLEMENTACIÓN

Se presentarán las siguientes opciones a eliminar conocimiento:

- Sentencias menos importantes a eliminar: se genera una lista con las sentencias menos importantes de cada subconjunto que deduce  $\alpha$ .  
 $\text{menosImportantes}(K, \text{EliminarMenosImp}).$
- Sentencias a eliminar, conservando las más importantes: Se genera una lista con las sentencias de cada subconjunto que deduce  $\alpha$ , menos las importantes.  
 $\text{masImportantes}(K, \text{ConservarMasImp}).$



- Menor cantidad de sentencias a eliminar: se genera una lista con las sentencias de cada subconjunto que deduce  $\alpha$ , buscando el menor conjunto posible.

`minimosDeCadaUno (K, EliminarMinCant).`

`opcionesContraccion ( Alpha,`

`EliminarMenosImp,`

`ConservarMasImp,`

`EliminarMinCant) :-`

`findall (K0, deduce (Alpha,K0),K),`

`listaNoVacia(K),`

`menosImportantes (K, EliminarMenosImp),`

`masImportantes (K, ConservarMasImp),`

`minimosDeCadaUno (K, EliminarMinCant).`

Para realizar la eliminación efectiva es necesario definir el predicado para eliminar el conocimiento de la BC.

`eliminarDeBC ([H|T]).`

Una vez definido lo anterior se definen los predicados para llevar a cabo la contracción:

`contraccionPorMenosImportantes(Alpha):-`

`findall(K0,deduce(Alpha,K0),K),`

`menosImportantes(K,Menos),`

`eliminarDeBC (Menos).`

`contraccionPorMantenerMasImportantes(Alpha):-`

`findall(K0,deduce(Alpha,K0),K),`

`masImportantes(K,Mas),`

`eliminarDeBC (Mas).`

```

contraccionPorMinimaCantidad(Alpha):-
    findall(K0,deduce(Alpha,K0),K),
    minimosDeCadaUno(K,Minimos),
    eliminarDeBC (Minimos).

```

---

## REVISAR(A)

El objetivo de realizar una operación de revisión priorizada es agregar un  $\alpha$  a la BC y que esta se mantenga consistente.

---

## DETALLE DE IMPLEMENTACIÓN

Revisar por los distintos criterios implica contraer dicho criterio por  $\neg \alpha$  y luego expandir por  $\alpha$ .

- Revisar por menor importante  

```
revisarPorMenosImportantes(Alpha)
```
- Revisar por mas importante  

```
revisarPorMantenerMasImportantes(Alpha)
```
- Revisar por mínima cantidad  

```
revisarPorMinimaCantidad(Alpha)
```

```

opcionesRevision (    Alpha,
                    EliminarMenosImp,
                    ConservarMasImp,
                    EliminarMinCant) :-
    opcionesContraccion(  no(Alpha),
                        EliminarMenosImp,
                        ConservarMasImp,
                        EliminarMinCant),
    !.

```

---

## CONSOLIDAR

Es un caso particular de la operación de contracción donde se eliminan inconsistencias de un estado de conocimiento. Por esa razón consolidar es igual a contraer por falso.

Se deben satisfacer las siguientes propiedades:

1. Inclusión
2. Consistencia
3. Relevancia y Retención de Núcleo.

---

## DETALLE DE IMPLEMENTACIÓN

Se presentarán las siguientes opciones a eliminar conocimiento:

- Sentencias menos importantes a eliminar.
- Sentencias a eliminar, conservando las más importantes.
- Menor cantidad de sentencias a eliminar.

```
opcionesConsolidar (      EliminarMenosImp,  
                          ConservarMasImp,  
                          EliminarMinCant).
```

Se definen los predicados para llevar a cabo la consolidación:

```
consolidarPorMinimaCantidad:-  
    contraccionPorMinimaCantidad(false).
```

```
consolidarPorMenosImportantes:-  
    contraccionPorMenosImportantes(false).
```

```
consolidarPorMantenerMasImportantes:-  
    contraccionPorMantenerMasImportantes(false).
```

---

## MOSTRAR

Imprime la BC.

---

## DETALLE DE IMPLEMENTACIÓN

Se utiliza el predicado findall/3 para generar una lista de resultados mostrando los kb(X) y las ee(Y,Z).

```
mostrarBC (KB,EE) :- findall (X, kb(X), KB),  
                    findall ([Y|Z],  
                             ee (Y,Z), EE).
```

## TUTORIAL

### HOME

Al abrir la aplicación web “Dinámica de Conocimiento” se encontrará con la página principal donde tendrá acceso a la aplicación y al código fuente.



### APLICACIÓN

Al acceder a la solapa aplicación, tendrá acceso a todas las operaciones disponibles para realizar el cambio de creencias.

Mediante la primera operación (expandir) podrá ingresar nuevas creencias a la base de conocimiento en conjunto con sus relaciones de importancia.

En caso de querer expandir por una relación compleja, deberá expandir por cada una de estas.

Por ejemplo

- Creencia:  $(a \vee b \rightarrow c)$
- Relación:  $(a \vee b \rightarrow c) \leq d$
- Creencia:  $(a \vee b \rightarrow c)$
- Relación:  $(a \vee b \rightarrow c) \leq b$

Para las operaciones contraer y revisar se deberá rellenar en el campo que identifica a la operación que se desea realizar y hacer clic en el botón correspondiente (Contraer o Revisar).

En caso de rellenar más de un campo en una misma operación, se realizaran en forma secuencial dichas operaciones (orden en el que se encuentran los campos).

Aplicación

192.168.1.71/prolog/aplicacion.php

Dinámica de conocimientoHomeAplicaciónCódigo Fuente

## Aplicación

Podrás realizar las operaciones de expandir (agregar una creencia a K), contraer (K en base a una creencia), hacer revisiones (sobre K respecto a una creencia), consolidaciones (devuelve la consistencia de K) y ver tu base de conocimiento en su estado actual por cada operación realizada.

Expandir

Creencia:

Relación de importancia a lo sumo tan importante:

Relación de importancia mas importante que:

Expandir

Contraer

Por menos importante:

Por mantener más importantes:

Por mínima cantidad:

Contraer

Revisar

Por menos importante:

Por mantener más importantes:

Por mínima cantidad:

Revisar

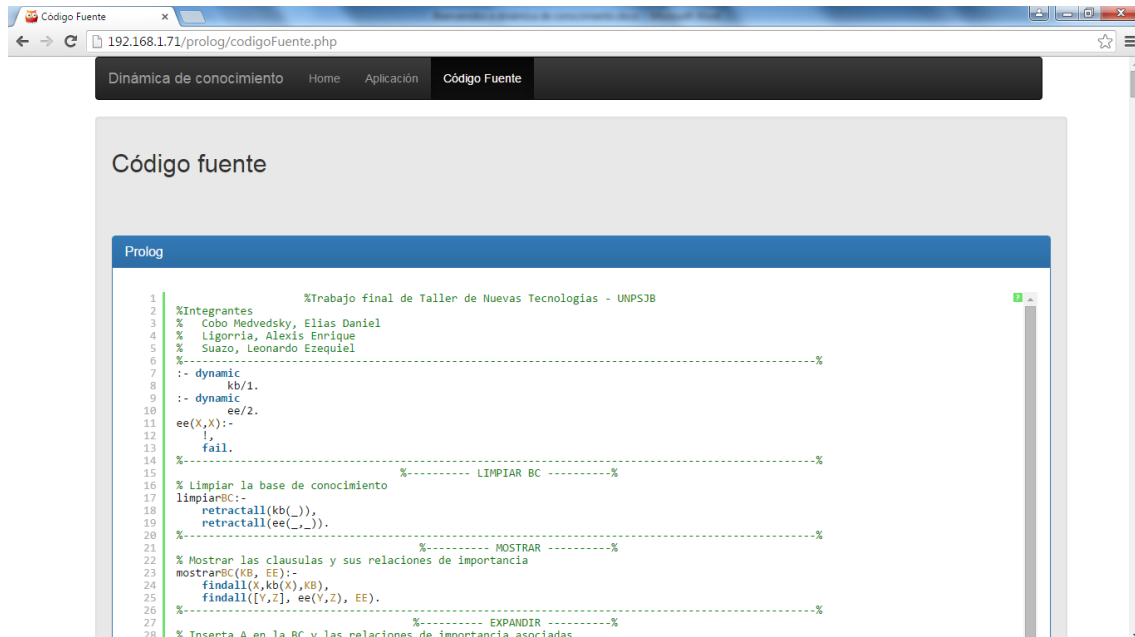
Consolidar

Seleccionar

Consolidar

COBO MEDVEDSKY, Elias Daniel - LIGORRIA, Alexis Enrique - SUAZO, Leonardo Ezequiel  
UNPSJB - Facultad de Ingeniería - Departamento de Informática - Taller de Nuevas Tecnologías

## CÓDIGO FUENTE



```
1 %Trabajo final de Taller de Nuevas Tecnologias - UNPSJB
2 %Integrantes
3 % Cobo Medvedsky, Elias Daniel
4 % Ligorria, Alexis Enrique
5 % Suazo, Leonardo Ezequiel
6 %-----
7 :- dynamic
8     kb/1.
9 :- dynamic
10    ee/2.
11 ee(X,X):-
12     !,
13     fail.
14 %-----
15 %----- LIMPIAR BC -----
16 % Limpiar la base de conocimiento
17 limpiarBC:-
18     retractall(kb(_)),
19     retractall(ee(_,_)).
20 %-----
21 %----- MOSTRAR -----
22 % Mostrar las clausulas y sus relaciones de importancia
23 mostrarBC(KB, EE):-
24     findall(X,kb(X),KB),
25     findall([V,Z], ee(V,Z), EE).
26 %-----
27 %----- EXPANDIR -----
28 % Inserta A en la RC y las relaciones de importancia asociadas
```



## EJEMPLOS

### EJEMPLO 1

$$Kb = \{ f, m, m \rightarrow f, f \rightarrow m \}$$

$$EE = f \leq m$$

$$f \leq f \rightarrow m$$

$$m \rightarrow f \leq m$$

### EXPANDIR BC

Expandir

Creencia:

Relación de importancia a lo sumo tan importante:

Relación de importancia mas importante que:

Expandir

### IMPRIMIR BC

Estado actual de la Base de Conocimiento

Última operación: expandirPorPHP(f->m,"").

KB= [ (f->m), (m->f),m,f]

EE= [[ (m->f),m],[f, (f->m)],[f,m]]

---

## CONTRAER POR MENOS IMPORTANTE

**Contraer**

**Por menos importante:**

**Por mantener más importantes:**

**Por mínima cantidad:**

---

## IMPRIMIR BC

**Estado actual de la Base de Conocimiento**

Última operación: mostrarOpcionesContraccionPHP(m-y-f),contraerPorMenosImportantePHP(m-y-f).

Eliminar por menos importante= [f, (m->f)]

Eliminar conservando mas importantes= [f, (m->f)]

Eliminar por mínima cantidad= [m, (f->m)]

KB= [ (f->m),m]

EE= []

---

## EXPANDIR E IMPRIMIR BC

**Estado actual de la Base de Conocimiento**

Última operación: expandirPorPHP(f,m,"").

KB= [f, (f->m),m]

EE= [[f,m]]

**Expandir**

**Creencia:**

**Relación de importancia a lo sumo tan importante:**

**Relación de importancia mas importante que:**

---

## CONSOLIDAR POR MENOS IMPORTANTE

**Consolidar**

---

## IMPRIMIR BC

Estado actual de la Base de Conocimiento

Última operación: mostrarOpcionesConsolidarPHP.consolidarPorMenosImportantePHP.

KB= {f, (f->m),m}

EE= {[f,m]}

## EJEMPLO 2

$Kb = \{ u \rightarrow n, n \rightarrow p, p \rightarrow s, s \rightarrow j, j \rightarrow b \}$

$EE = u \rightarrow n \leq n \rightarrow p$

$n \rightarrow p \leq p \rightarrow s$

$p \rightarrow s \leq s \rightarrow j$

$s \rightarrow j \leq j \rightarrow b$

---

## EXPANDIR E IMPRIMIR BC

Estado actual de la Base de Conocimiento

Última operación: expandirPorPHP(s->j,j->b,"").

KB= { (s->j), (p->s), (n->p), (u->n) }

EE= {[ (s->j), (j->b) ], [ (s->j), (p->s) ], [ (n->p), (p->s) ], [ (u->n), (n->p) ] }

Expandir

Creencia:

Relación de importancia a lo sumo tan importante:

Relación de importancia mas importante que:

Expandir

---

## REVISAR POR MÍNIMA CANTIDAD

Estado actual de la Base de Conocimiento

Última operación: mostrarOpcionesRevisarPHP(no((s->j))),revisarPorMinimaCantidadPHP(no((s->j))).

KB= [no((s->j)), (p->s), (n->p), (u->n)]

EE= {[ (n->p), (p->s) ], [ (u->n), (n->p) ] }

---

## CONTRAER POR MANTENER MAS IMPORTANTES

Contraer

Por menos importante:

Por mantener más importantes:

$(n \rightarrow p)$

Por mínima cantidad:

Contraer

---

## IMPRIMIR BC

Estado actual de la Base de Conocimiento

Última operación: mostrarOpcionesContraccionPHP((n->p)),contraerPorMantenerMasImportantesPHP((n->p)).

KB= [no((s->j)), (p->s), (u->n)]

EE= []

## EJEMPLO 3

$$Kb = \{ (a \vee b) \rightarrow c, b, d, a, d \rightarrow c, e, f \}$$

$$\begin{aligned} EE = & (a \vee b) \rightarrow c \leq e \\ & (a \vee b) \rightarrow c \leq d \rightarrow a \\ & d \rightarrow c \leq e \\ & d \rightarrow c \leq b \\ & d \rightarrow a \leq f \\ & f \leq d \end{aligned}$$

---

## EXPANDIR E IMPRIMIR BC

Estado actual de la Base de Conocimiento
Última operación: expandirPorPHP(d->a,"").
KB= [ (d->a),e,d,b,f, (d->c), (a-o-b->c)]
EE= [[f,d],[ (d->c),f],[ (d->c),b],[ (d->c),e],[ (a-o-b->c), (d->a)],[ (a-o-b->c),e]]

---

## CONTRAER POR MENOS IMPORTANTE

Contraer
Por menos importante:
<input type="text" value="e"/>
Por mantener más importantes:
<input type="text"/>
Por mínima cantidad:
<input type="text"/>
<input type="button" value="Contraer"/>

---

## IMPRIMIR BC

Estado actual de la Base de Conocimiento
Última operación: mostrarOpcionesContraccionPHP(e),contraerPorMenosImportantePHP(e).
KB= [ (d->a),d,b,f, (d->c), (a-o-b->c)]
EE= [[f,d],[ (d->c),f],[ (d->c),b],[ (a-o-b->c), (d->a)]]

---

## REVISAR POR MENOS IMPORTANTE

Revisar
Por menos importante:
<input type="text" value="no(d)"/>
Por mantener más importantes:
<input type="text"/>
Por mínima cantidad:
<input type="text"/>
<input type="button" value="Revisar"/>

---

## IMPRIMIR BC

Estado actual de la Base de Conocimiento

Última operación: mostrarOpcionesRevisarPHP(no(d)),revisarPorMenosImportantePHP(no(d)).  
KB= [no(d), (d->a),b,f, (d->c), (a-o-b->c)]  
EE= [[ (d->c),f],[ (d->c),b],[ (a-o-b->c), (d->a)]]

---

## EXPANDIR E IMPRIMIR BC (INCONSISTENCIA)

Expandir

Creencia:  
  
Relación de importancia a lo sumo tan importante:  
  
Relación de importancia mas importante que:

Estado actual de la Base de Conocimiento

Última operación: expandirPorPHP(d,"").  
KB= [d,no(d), (d->a),b,f, (d->c), (a-o-b->c)]  
EE= [[ (d->c),f],[ (d->c),b],[ (a-o-b->c), (d->a)]]

---

## CONSOLIDAR POR MÍNIMA CANTIDAD

Consolidar

Estado actual de la Base de Conocimiento

Última operación: mostrarOpcionesConsolidarPHP,consolidarPorMinimaCantidadPHP.  
KB= [no(d), (d->a),b,f, (d->c), (a-o-b->c)]  
EE= [[ (d->c),f],[ (d->c),b],[ (a-o-b->c), (d->a)]]

## BIBLIOGRAFÍA

- [http://programacion.net/articulo/curso\\_avanzado\\_de\\_prolog\\_166/3](http://programacion.net/articulo/curso_avanzado_de_prolog_166/3)
- [http://sedici.unlp.edu.ar/bitstream/handle/10915/21444/Documento\\_completo.pdf?sequence=1](http://sedici.unlp.edu.ar/bitstream/handle/10915/21444/Documento_completo.pdf?sequence=1)
- [http://sedici.unlp.edu.ar/bitstream/handle/10915/23085/Documento\\_completo.pdf?sequence=1](http://sedici.unlp.edu.ar/bitstream/handle/10915/23085/Documento_completo.pdf?sequence=1)