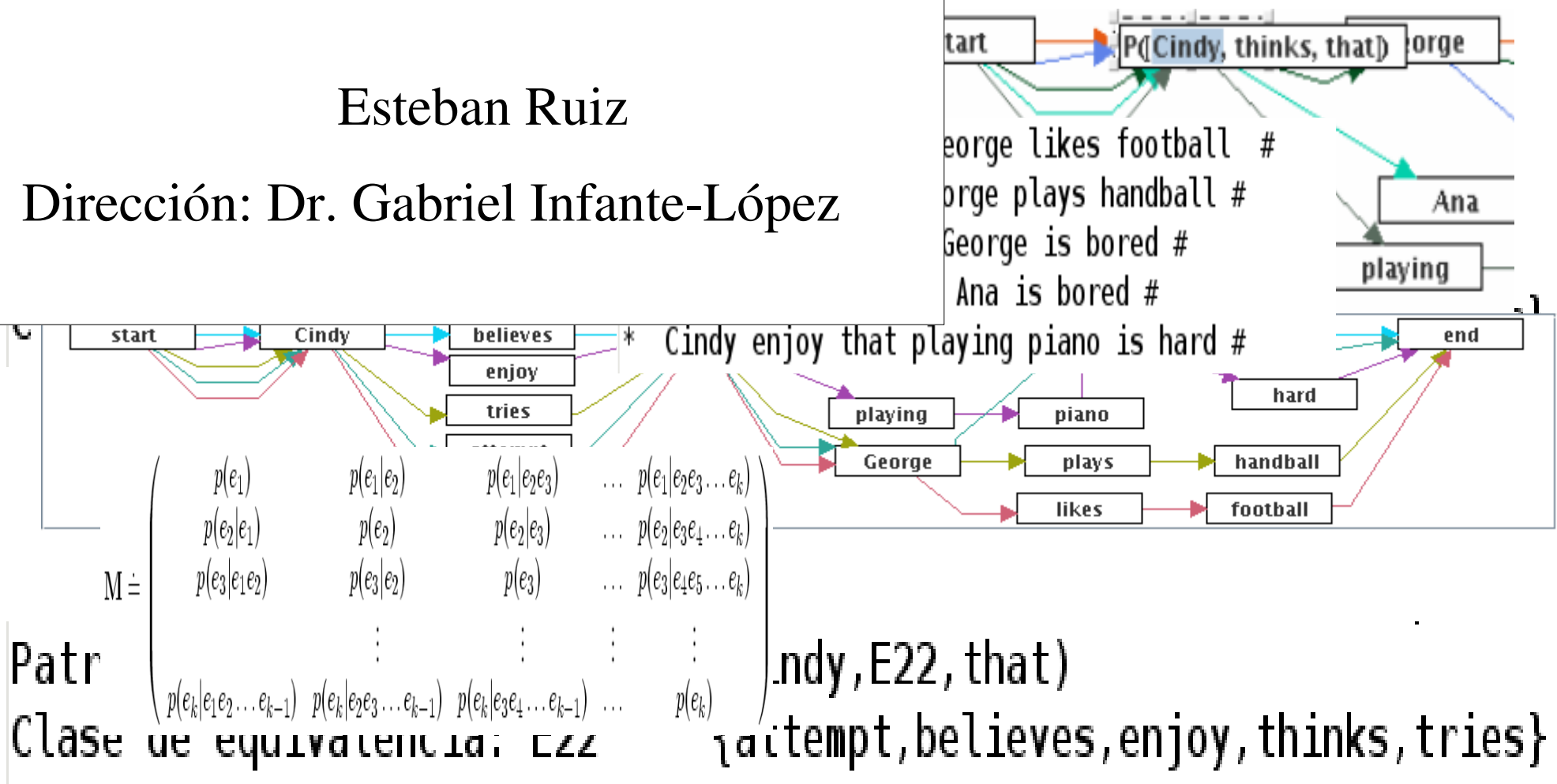
El algoritmo ADIOS

Esteban Ruiz

Dirección: Dr. Gabriel Infante-López



Contenido

- ¿Qué es?
- Ventajas
- Ideas generales
- Conceptos útiles
- El procedimiento MEX, caminos generalizados
- Esquema del algoritmo
- Dificultades e implementación. Aplicaciones

¿Qué es?

El problema:

Inferir reglas subyacentes en corpus no anotados.

ADIOS: Automatic Distillation of Structure

Z. Solan, D. Horn, E. Ruppin, S. Edelman (TAU)

```
*  Cindy thinks that George likes football  #
*  Cindy tries that George plays handball #
*  Cindy attempt that George is bored #
*  Cindy believes that Ana is bored #
*  Cindy enjoy that playing piano is hard #
```

```
Patrón:                    P21     (Cindy,E22,that)
Clase de equivalencia: E22     {attempt,believes,enjoy,thinks,tries}
```

Ventajas y características

Ventajas y características

- No supervisado
- Corpus no estructurado
- Combina probabilidades y reglas

Desventajas

- Infiere sólo gramáticas limitadas

Ideas generales del algoritmo

Corpus, léxico, símbolos especiales

```
* Cindy thinks that George likes footba
* Cindy tries that George plays
* Cindy attempt that George is b
* Cindy believes that Ana is bor
* Cindy enjoy that playing piano
```

Léxico

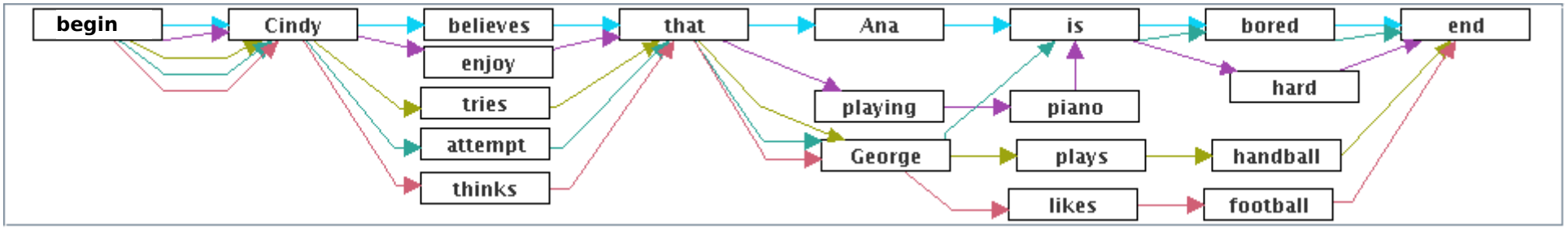
Ana	hard
Cindy	is
George	likes
attempt	piano
believes	playing
bored	plays
enjoy	that
football	thinks
handball	tries

Dos símbolos especiales:

begin

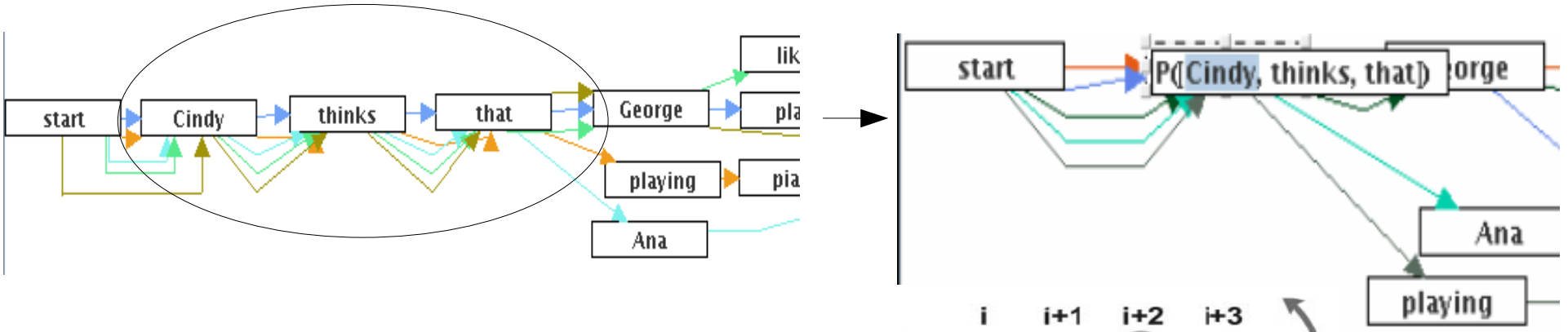
end

Cargar el corpus en un **pseudografo**



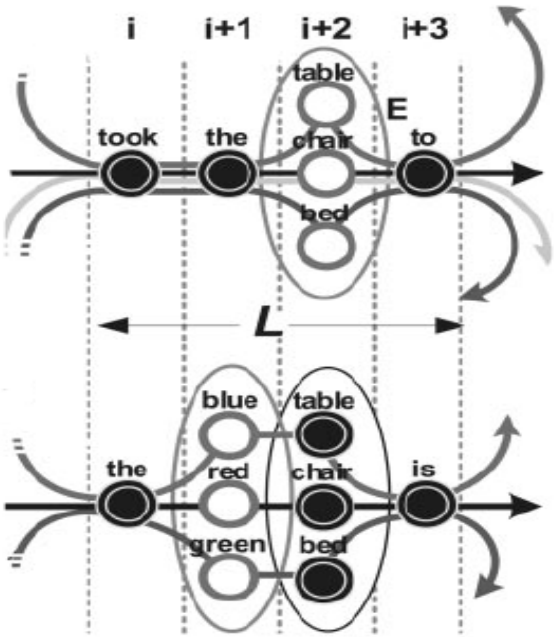
Ideas generales del algoritmo

En cada camino:
Detección de patrones y reescritura del grafo

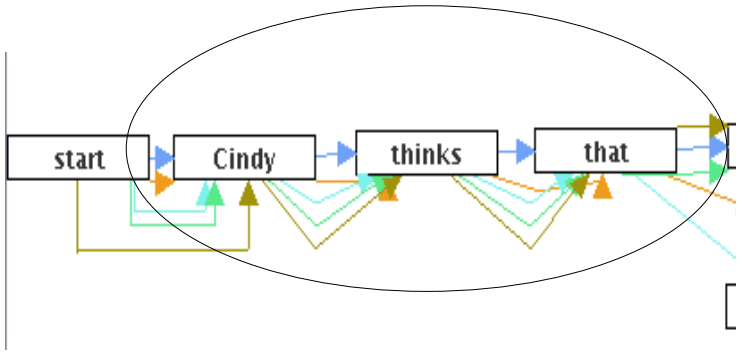


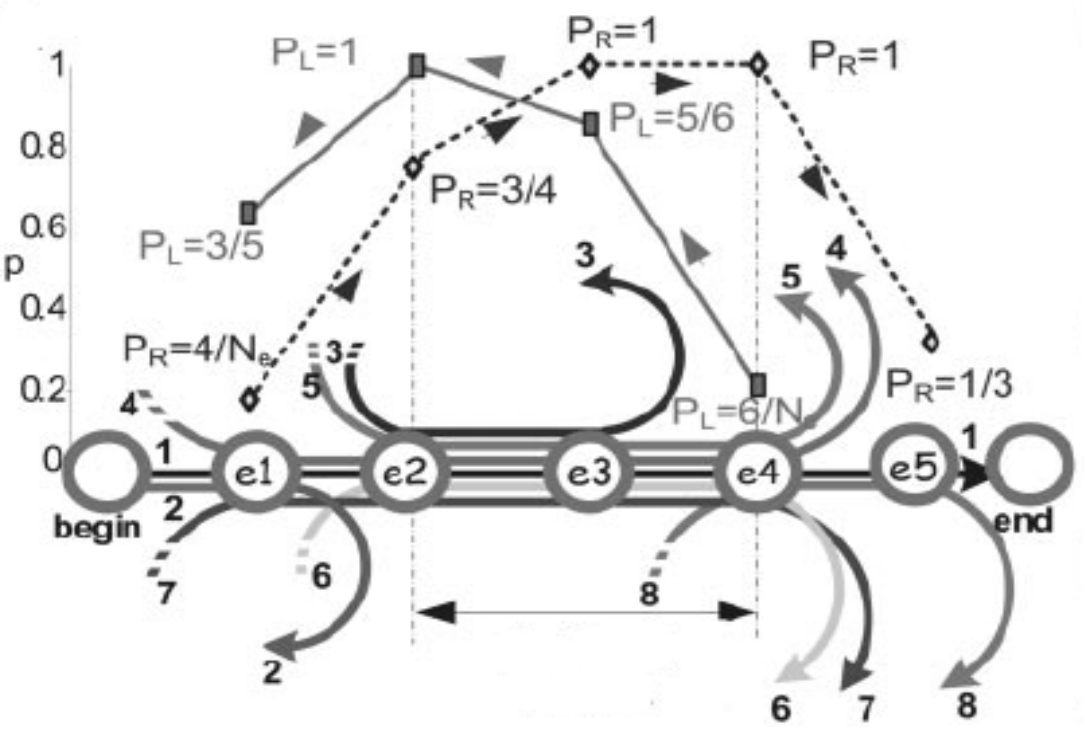
Detección de patrones más complejos:

clases de equivalencia y caminos generalizados



Conceptos útiles





Definición de P_R y P_L

Dado un camino de búsqueda $S = (e_1 \rightarrow e_2 \rightarrow ... \rightarrow e_k) = (e_1; e_k)$

$$P_R(e_i; e_j) = p(e_j | e_i e_{i+1} e_{i+2} ... e_{j-1}) = \frac{l(e_i; e_j)}{l(e_i; e_{j-1})} \text{ (si } i < j)$$

$$P_R(e_i; e_i) = \frac{l(e_i)}{\sum_{x=0}^{N} l(e_x)}$$

IDEM p/ P_L

Más conceptos útiles

- Matriz M

$$M_{ij}(\mathbf{S}) = \begin{cases} P_R(e_i; e_j) & \text{if } i > j \\ P_L(e_j; e_i) & \text{if } i < j \\ P(e_i) & \text{if } i = j. \end{cases}$$

$$\mathrm{M} \doteq \begin{pmatrix} p(e_1) & p(e_1|e_2) & p(e_1|e_2e_3) & \dots & p(e_1|e_2e_3\dots e_k) \\ p(e_2|e_1) & p(e_2) & p(e_2|e_3) & \dots & p(e_2|e_3e_4\dots e_k) \\ p(e_3|e_1e_2) & p(e_3|e_2) & p(e_3) & \dots & p(e_3|e_4e_5\dots e_k) \\ & \vdots & \vdots & \vdots & \vdots \\ p(e_k|e_1e_2\dots e_{k-1}) & p(e_k|e_2e_3\dots e_{k-1}) & p(e_k|e_3e_4\dots e_{k-1}) & \dots & p(e_k) \end{pmatrix}$$

- D_R y D_L (Relaciones de decrecimiento)

$$D_R(e_i; e_j) = P_R(e_i; e_j)/P_R(e_i; e_{j-1})$$

- Prueba de significación

$$B(e_i; e_j) = \sum_{x=0}^{l(e_i;e_j)} Binom(x, l(e_i; e_{j-1}), \eta P_R(e_i; e_{j-1})) < \alpha; \alpha \ll 1.$$

El procedimiento MEX (simplificado)

Sea p el camino a analizar

- Calcular P_R y P_L para cada subcamino $e_i \rightarrow .. \rightarrow e_j$ de p
- Construir *Dr_candidatos*: por cada comienzo posible e_i de un subcamino
  - Por cada final posible e_j de ese subcamino:
    - Si $D_R(e_i; e_j) < \eta$ y la prueba de significación indica que la muestra es significante entonces marcar ese par como un sección candidata.
- Hacer lo mismo de derecha a izquierda (calculo de D_L significantes).
- Buscar las secciones candidatas que pueden definir un patrón: si $D_R(a, b)$ y $D_L(d, c)(c < d)$ son secciones candidatas, deben cumplir:

$$d \geq b - 1 \wedge c \geq a - 1 \wedge c < b - 2 \wedge \neg(c < 0 \wedge b \geq \#p - 1)$$

- Retornar el patrón candidato con menor significación

Esquema del algoritmo:

- Inicialización
- Destilación de patrones
- Generalización: primer paso
- Generalización: bootstrap (repetir)

Inicializacion:

Repetir hasta el

fin del archivo:

- Leer el archivo hasta encontrar el final de una secuencia
- Cargar los símbolos encontrados como un nuevo camino en el pseudografo

Destilación de patrones

Con cada camino:

- Ejecutar MEX en ese camino
- Si se obtuvo un patrón reescribir (rewire) el grafo

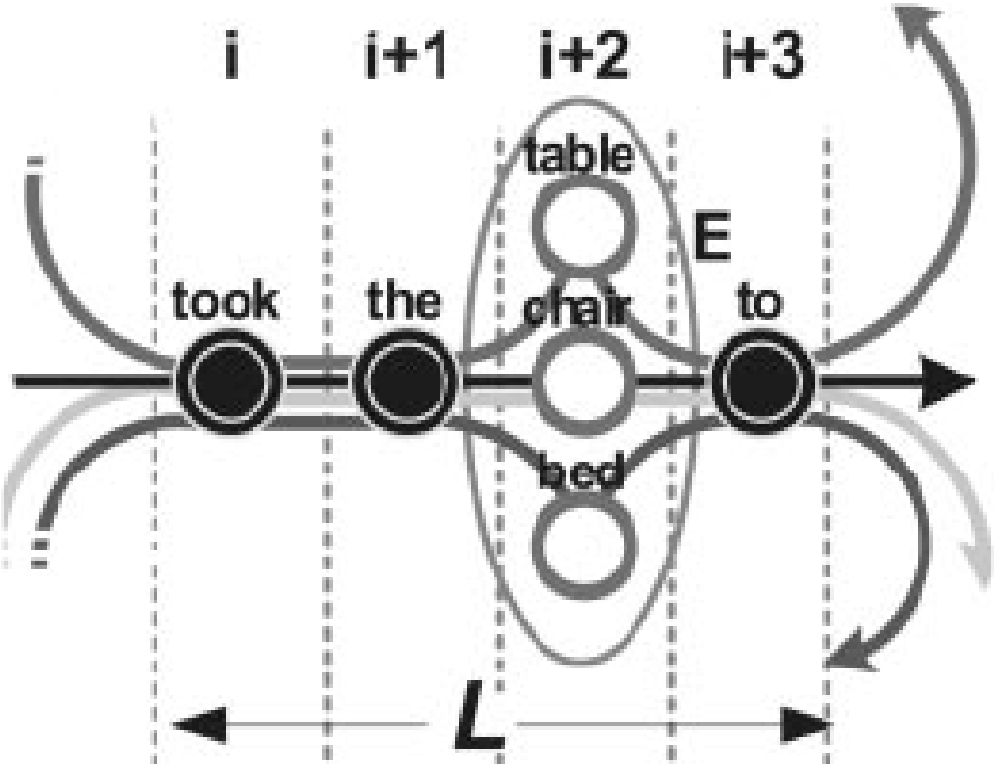
Esquema del algoritmo:

- Inicialización
- Destilación de patrones
- Generalización: primer paso
- Generalización: bootstrap (repetir)

Gen: primer paso

Con cada camino:

- Por cada posición posible de una ventana de largo L:
  - Considerar todos los huecos posibles en esa ventana y ejecutar MEX para cada caso
- Seleccionar el mejor patrón encontrado y reescribir el grafo (nueva clase de equiv)



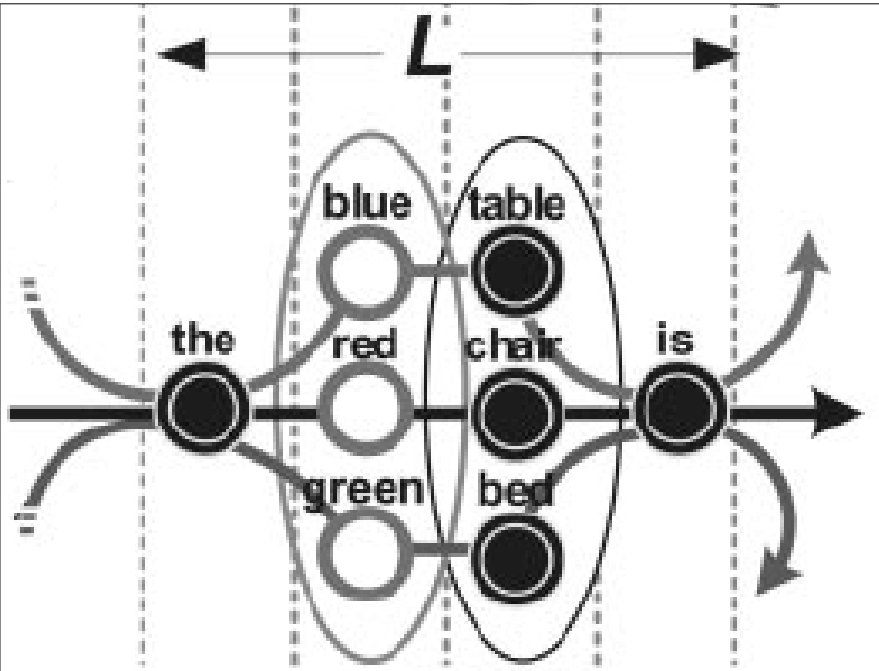
Esquema del algoritmo:

- Inicialización
- Destilación de patrones
- Generalización: primer paso
- Generalización: bootstrap (repetir)

Gen: bootstrap

Con cada camino:

- Con cada posición de una ventana de largo L
  - Construir el camino generalizado
  - Reducir el camino generalizado
  - Realizar MEX sobre el camino generalizado reducido
- Si se detectó un patrón:
  - ¿nueva clase de equiv?
  - Reescribir el grafo



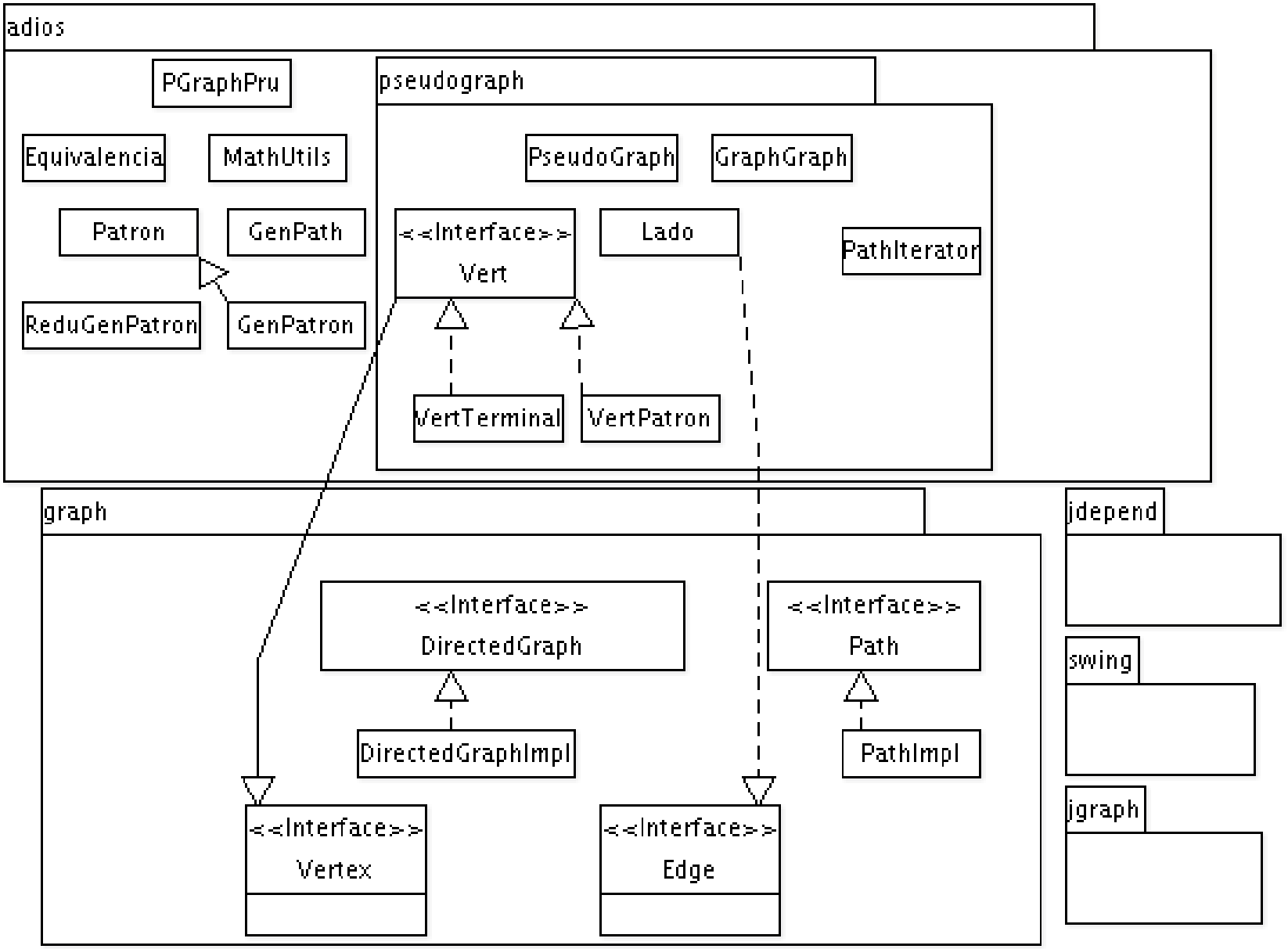
Dificultades

- Calculo de la binomial
- Especificación del algoritmo
- Definición de camino generalizado
- Adios-lite
- Prueba de significación

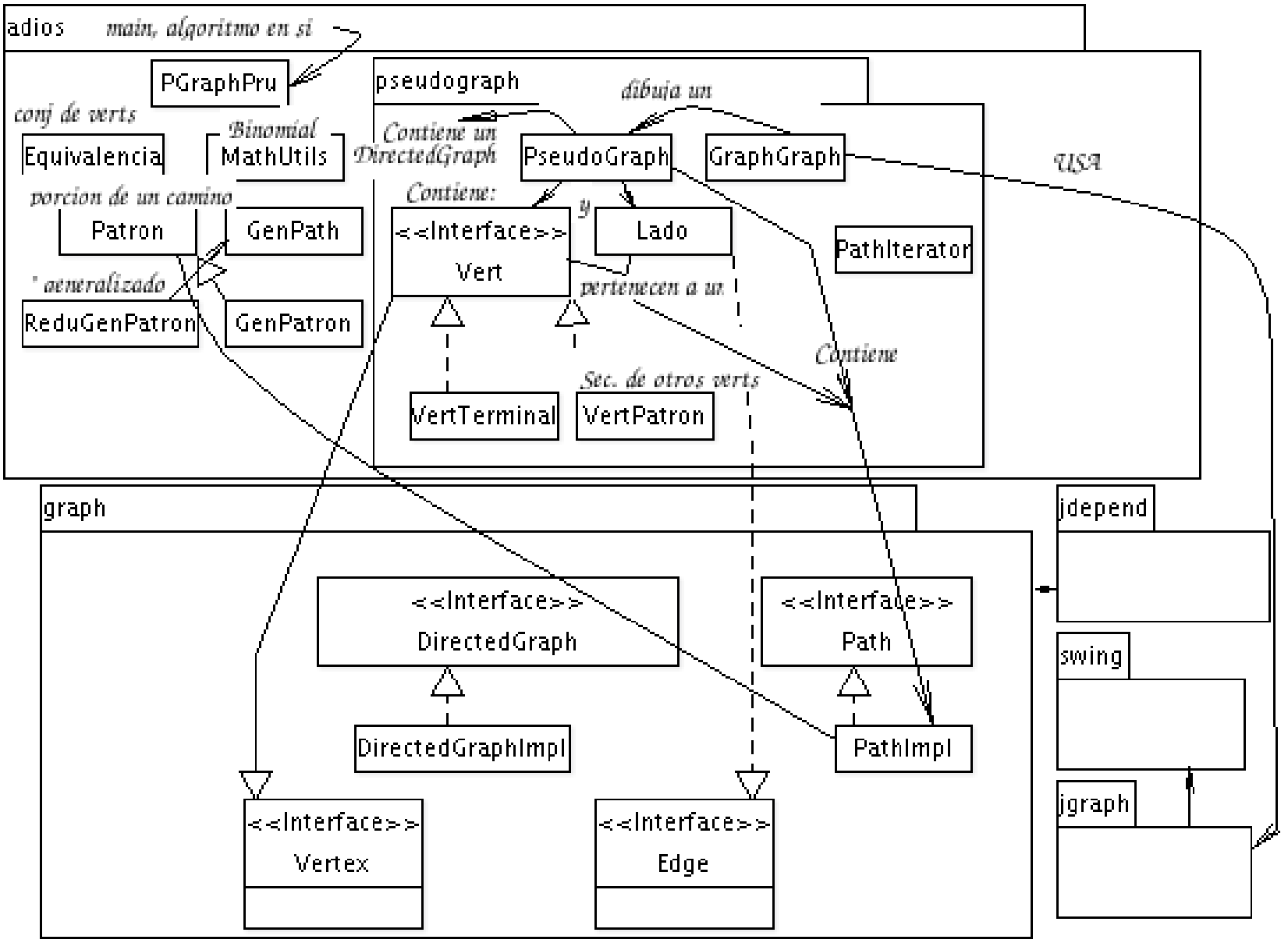
Software utilizado

- cvs
- eclipse
- JDK 6
- Librerias de apache y Jgraph
- ArgoUML

Diseño


adios
PGraphPru
pseudograph
Equivalencia
MathUtils
PseudoGraph
GraphGraph
Patron
GenPath
<<Interface>>
Vert
Lado
PathIterator
ReduGenPatron
GenPatron
VertTerminal
VertPatron
graph
<<Interface>>
DirectedGraph
<<Interface>>
Path
DirectedGraphImpl
PathImpl
<<Interface>>
Vertex
<<Interface>>
Edge
jdepend
swing
jgraph

Diseño


adios
main, algoritmo en si
PGraphPru
pseudograph
dibuja un
conj de verts
Binomial
Contiene un
DirectedGraph
Equivalencia
MathUtils
PseudoGraph
GraphGraph
USA
porcion de un camino
Contiene:
y
Patron
GenPath
<<Interface>>
Vert
Lado
PathIterator
" generalizado
pertenecen a un
ReduGenPatron
GenPatron
Contiene
Sec. de otros verts
VertTerminal
VertPatron
graph
jdepend
<<Interface>>
DirectedGraph
<<Interface>>
Path
swing
DirectedGraphImpl
PathImpl
jgraph
<<Interface>>
Vertex
<<Interface>>
Edge

Experiencias de la implementación

- Algo de documentación
- Testing
- Problemas con la especificación
- Resultados del diseño OO

Bibliografía

Z. Solan, D. Horn, E. Ruppin and S. Edelman, Unsupervised learning of natural languages. Editado por James L. McClelland, Carnegie Mellon University, Pittsburgh, PA, y aprobado June 14, 2005.

D. S. Moore, Estadística aplicada básica, Antoni Bosch editor, 1995.

M. Triola, Estadística elemental, Addison Wesley Longman, 7ma. ed., 2000.

D. K. Hildebrand, L. Ott, Estadística aplicada a la administración y a la economía, Addison Wesley Longman, 3ra. ed, 1998.

J. Makkonen, H. Ahonen-Myka and Marko Salmenkivi, Applying Semantic Classes in Event Detection and Tracking.

J. Weeds, D. Weir and D. McCarthy, Characterising Measures of Lexical Distributional Similarity.

Bibliografía

J. Brookshear, Lenguajes formales, autómatas y complejidad. Addison-Wesley Iberoamericana.

L. A. Ballesteros, Resolving ambiguity for cross-language information retrieval: A dictionary approach, Univ. of Massachusetts, 2001.

N. K. Bosa, P. Liang, Neural Network Fundamentals with Graphs, Algorithms, and Applications.

P. G. Hoel, S. C. Port, C. J. Stone, Introduction to Stochastic Proceses, Waveland Press, 1987.

C. M. Grinstead, J. L. Snell, Introduction to Probability, AMS, second revised edition, 1997.

G. Infante Lopez, Two level grammars for natural language parsing, Soluciones Gráficas, 2005.

Agradecimientos y preguntas

ADIOS