Question answering de dominio abierto y de dominio cerrado

Julián Peller

Abril 2016

index.tex

index.tex

index.tex

index.tex

index.tex

¿Qué es question answering?

Question Answering

Es el proceso automatizado de generación de respuestas concretas para preguntas formuladas en lenguaje natural.

Question

¿Quién desarrolló la teoría de la relatividad?

Answer

Albert Einstein.

¿Qué es question answering?

Question Answering

Es el proceso automatizado de generación de respuestas concretas para preguntas formuladas en lenguaje natural.

Question

¿Quién desarrolló la teoría de la relatividad?

Answer

Albert Einstein.

¿Qué es question answering?

Question Answering

Es el proceso automatizado de generación de respuestas concretas para preguntas formuladas en lenguaje natural.

Question

¿Quién desarrolló la teoría de la relatividad?

Answer

Albert Einstein.

¿Qué es question answering?

Question Answering

Es el proceso automatizado de generación de respuestas concretas para preguntas formuladas en lenguaje natural.

Question

¿Quién desarrolló la teoría de la relatividad?

Answer

Albert Einstein.

Clasificación

Generalidad del dominio

- Dominio abierto o dominio cerrado

Tipo de datos

- Estructurados (Bases de datos)
- No estructurados (Colecciones de textos)

Soporte de preguntas

- Fácticas, listas, definiciones, modo, razón
- Cláusulas de tiempo, sin respuesta, etc.

Clasificación

Generalidad del dominio

- Dominio abierto o dominio cerrado

Tipo de datos

- Estructurados (Bases de datos)
- No estructurados (Colecciones de textos)

Soporte de preguntas

- Fácticas, listas, definiciones, modo, razón
- Cláusulas de tiempo, sin respuesta, etc.

Clasificación

Generalidad del dominio

- Dominio abierto o dominio cerrado

Tipo de datos

- Estructurados (Bases de datos)
- No estructurados (Colecciones de textos)

Soporte de preguntas

- Fácticas, listas, definiciones, modo, razón
- Cláusulas de tiempo, sin respuesta, etc.

Clasificación

Generalidad del dominio

- Dominio abierto o dominio cerrado

Tipo de datos

- Estructurados (Bases de datos)
- No estructurados (Colecciones de textos)

Soporte de preguntas

- Fácticas, listas, definiciones, modo, razón
- Cláusulas de tiempo, sin respuesta, etc.

index.tex

Qué es esta tesis

Brevemente

Implementamos dos sistemas de question answering más o menos decentes.

Dominio cerrado con soporte para inglés / Popescu World

- Base de datos relacional de juguete
- Modelo teórico de *tratabilidad semántica*
- Traduce preguntas a consultas SQL

Dominio abierto con soporte multilingüe / Multilingual Qanus

- Wikipedia(s) como base de conocimientos
- Framework Qanus y librería Freeling
- Basado en IR + NLP + heurísticas

Qué es esta tesis

Brevemente

Implementamos dos sistemas de question answering más o menos decentes.

Dominio cerrado con soporte para inglés / Popescu World

- Base de datos relacional de juguete
- Modelo teórico de *tratabilidad semántica*
- Traduce preguntas a consultas SQL

Dominio abierto con soporte multilingüe / Multilingual Qanus

- Wikipedia(s) como base de conocimientos
- Framework Qanus y librería Freeling
- Basado en IR + NLP + heurísticas

Qué es esta tesis

Brevemente

Implementamos dos sistemas de question answering más o menos decentes.

Dominio cerrado con soporte para inglés / Popescu World

- Base de datos relacional de juguete
- Modelo teórico de *tratabilidad semántica*
- Traduce preguntas a consultas SQL

Dominio abierto con soporte multilingüe / Multilingual Qanus

- Wikipedia(s) como base de conocimientos
- Framework Qanus y librería Freeling
- Basado en IR + NLP + heurísticas

Qué es esta tesis

Brevemente

Implementamos dos sistemas de question answering más o menos decentes.

Dominio cerrado con soporte para inglés / Popescu World

- Base de datos relacional de juguete
- Modelo teórico de *tratabilidad semántica*
- Traduce preguntas a consultas SQL

Dominio abierto con soporte multilingüe / Multilingual Qanus

- Wikipedia(s) como base de conocimientos
- Framework Qanus y librería Freeling
- Basado en IR + NLP + heurísticas

index.tex

index.tex

Introducción

- Sistema que implementa el modelo propuesto por Popescu en [Popescu et al. 2003a] y [Popescu et al. 2003b].
- Define concepto de tratabilidad semántica y traduce preguntas semánticamente tratables en consultas SQL
- Código implementado en java, accesible en http://github.com/julian3833/popescu-world
- Testeado sobre base de datos relacional de juguete provista por MySQL, llamada World, con información geográfica básica de países, ciudades e idiomas.
- Sistema de funcionalidad acotada con soporte solo para el inglés.

index.tex

Dominio de problemas

Dominio acotado y específico

- Restaurants, bancos, vuelos, libros, etcétera (pero solo uno)
- Bases de datos relacionales $\rightarrow$ datos estructurados
- Question answering como interfaz para una base de datos
- Traducir pregunta a SQL

Ejemplo

Pregunta

When did Albert Einstein die?

Consulta SQL

SELECT death_date **FROM** scientists
WHERE name = 'Albert Einstein'

Respuesta

April 18th, 1955

Ejemplo

Pregunta

When did Albert Einstein die?

Consulta SQL

SELECT death_date **FROM** scientists
WHERE name = 'Albert Einstein'

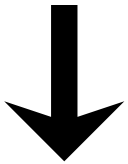
Respuesta

April 18th, 1955

Ejemplo

Pregunta

When did Albert Einstein die?



Consulta SQL

SELECT death_date **FROM** scientists
WHERE name = 'Albert Einstein'

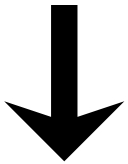
Respuesta

April 18th, 1955

Ejemplo

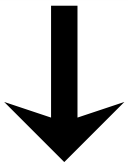
Pregunta

When did Albert Einstein die?



Consulta SQL

SELECT death_date **FROM** scientists
WHERE name = 'Albert Einstein'



Respuesta

April 18th, 1955

Modelo teórico

- Popescu et al. $\rightarrow$ QADB, Precise
- Tratabilidad semántica de una pregunta q en el contexto de una DB d.
- Las preguntas no tratables se rechazan, las tratables se traducen.
- Motivación: no se puede fallar activamente (mal interpretar)

Tratabilidad semántica / motivación

- La complejidad de las preguntas en lenguaje natural es arbitrariamente grande.
- Las preguntas semánticamente tratables son preguntas fáciles pero abarcativas
- Distinguir un subconjunto 1) *tratable* y 2) *abarcador*
- Rechazar y pedir reformulación de las no tratables
  - Es mejor no dar respuesta a dar una mala. Conservar la confianza en el sistema.

Tratabilidad semántica / idea coloquial

Tratabilidad semántica en el contexto de una DB

- Una Q-word (Qué, quién, cuándo, dónde, etc.)
- Pares de atributos y valores
- Valores sueltos
- Términos no significativos y menciones a relaciones
- Por ejemplo:
  - ¿Qué bancos de la empresa Credicoop están localizados en el barrio de Coghlan?
  - ¿Quién era el presidente de México en 1993?

Definiciones (1): Elemento, qword, compatibilidad, token, marcador sintáctico

- Elementos de una DB: Relaciones, Atributos y Valores
- Qwords - pronombres interrogativos
  - {What, where, which, when, who}
  - {Qué, dónde, cuál, cuándo, quién}
- Compatibilidad
  - Valor con sus atributo
  - Atributos con sus relaciones
  - Valor con las relaciones de sus atributos
  - Q-words compatibles con atributoes
    - Definición a mano especifica por DB
- Token: un conjunto de lemas de palabras de la pregunta que corresponden a un elemento de la base de datos.
  - Por ejemplo, {experiencia, requerir} y {experiencia, necesario} son dos tokens para "Experiencia Requerida"
- Marcador sintáctico: palabras que no aportan a la interpretación de la pregunta, definidas a mano

Definiciones (2): Correspondencia entre tokens y elementos

- Cada elemento de la base de datos se separa en palabras individuales:
  - "Experiencia Requerida" $\rightarrow$ {Experiencia, Requerida}
- Se genera un conjunto de sinónimos para cada palabra usando Wordnet:
  - experiencia $\rightarrow$ {experiencia, conocimiento, habilidad,...}
  - requerida $\rightarrow$ {requerida, necesaria, indispensable,...}
- Se toman los lemas o raíces de todas las palabras
  - experiencia $\rightarrow$ {experiencia, conocimiento, habilidad,...}
  - requerida $\rightarrow$ {requerir, necesario, indispensable,...}
- Se generan tokens combinando los lemas de los sinónimos de cada elemento:
  - tokens = {(experiencia, requerir), (conocimiento, requerir), (habilidad, requerir),(experiencia, necesario), (conocimiento, necesario), (habilidad, necesario),(experiencia, indispensable), (conocimiento, indispensable), (habilidad, indispensable)}
- Este conjunto de tokens son los tokens que se corresponden con el elemento
- Un token tiene tipos dependiendo a qué elementos corresponda

Definiciones (3): tokenización completa, asociación sintáctica

- **Tokenización completa de una pregunta *q***: cualquier conjunto de tokens en los que cada término que no es un marcador sintáctico de *q* aparece en exactamente un token del conjunto.
  - Una partición de tokens (como fueron definidos) de la pregunta *q*
- **Asociación sintáctica**: dos tokens están sintácticamente asociados en *q* si cumplen ciertas restricciones sintácticas.
  - Modelado con una función *attachment*(t1, t2, q) $\rightarrow$ *booleano*

Definiciones (4): traducción válida

Una **traducción válida** es un mapeo de una tokenización completa de q en elementos de E que cumple 3 condiciones:

1. Cada token se corresponde con un único elemento de E.
2. Cada token de atributo se relaciona con un único token de valor, cumpliendo que:
   - el atributo de la base de datos que corresponde al token de atributo y el valor de la base de datos que corresponde al token de valor son compatibles
   - ambos tokens están sintácticamente asociados
3. Cada token de relación está relacionado a un token de atributo o bien a un token de valor, cumpliendo las siguientes condiciones:
   - la relación de la base de datos que corresponde al token de relación y el elemento de la base de datos que corresponde al token de atributo o token de valor son compatibles
   - ambos tokens (token de relación - token de atributo o bien token de relación - token de valor) están sintácticamente asociados

Definiciones (4): traducción válida

Una **traducción válida** es un mapeo de una tokenización completa de q en elementos de E que cumple 3 condiciones:

1. Cada token se corresponde con un único elemento de E.
2. Cada token de atributo se relaciona con un único token de valor, cumpliendo que:
   - el atributo de la base de datos que corresponde al token de atributo y el valor de la base de datos que corresponde al token de valor son compatibles
   - ambos tokens están sintácticamente asociados
3. Cada token de relación está relacionado a un token de atributo o bien a un token de valor, cumpliendo las siguientes condiciones:
   - la relación de la base de datos que corresponde al token de relación y el elemento de la base de datos que corresponde al token de atributo o token de valor son compatibles
   - ambos tokens (token de relación - token de atributo o bien token de relación - token de valor) están sintácticamente asociados

Definiciones (4): traducción válida

Una **traducción válida** es un mapeo de una tokenización completa de q en elementos de E que cumple 3 condiciones:

1. Cada token se corresponde con un único elemento de E.
2. Cada token de atributo se relaciona con un único token de valor, cumpliendo que:
   - el atributo de la base de datos que corresponde al token de atributo y el valor de la base de datos que corresponde al token de valor son compatibles
   - ambos tokens están sintácticamente asociados
3. Cada token de relación está relacionado a un token de atributo o bien a un token de valor, cumpliendo las siguientes condiciones:
   - la relación de la base de datos que corresponde al token de relación y el elemento de la base de datos que corresponde al token de atributo o token de valor son compatibles
   - ambos tokens (token de relación - token de atributo o bien token de relación - token de valor) están sintácticamente asociados

Definiciones (4): traducción válida

Una **traducción válida** es un mapeo de una tokenización completa de q en elementos de E que cumple 3 condiciones:

1. Cada token se corresponde con un único elemento de E.
2. Cada token de atributo se relaciona con un único token de valor, cumpliendo que:
   - el atributo de la base de datos que corresponde al token de atributo y el valor de la base de datos que corresponde al token de valor son compatibles
   - ambos tokens están sintácticamente asociados
3. Cada token de relación está relacionado a un token de atributo o bien a un token de valor, cumpliendo las siguientes condiciones:
   - la relación de la base de datos que corresponde al token de relación y el elemento de la base de datos que corresponde al token de atributo o token de valor son compatibles
   - ambos tokens (token de relación - token de atributo o bien token de relación - token de valor) están sintácticamente asociados

Definiciones (4): traducción válida

Una **traducción válida** es un mapeo de una tokenización completa de q en elementos de E que cumple 3 condiciones:

1. Cada token se corresponde con un único elemento de E.
2. Cada token de atributo se relaciona con un único token de valor, cumpliendo que:
   - el atributo de la base de datos que corresponde al token de atributo y el valor de la base de datos que corresponde al token de valor son compatibles
   - ambos tokens están sintácticamente asociados
3. Cada token de relación está relacionado a un token de atributo o bien a un token de valor, cumpliendo las siguientes condiciones:
   - la relación de la base de datos que corresponde al token de relación y el elemento de la base de datos que corresponde al token de atributo o token de valor son compatibles
   - ambos tokens (token de relación - token de atributo o bien token de relación - token de valor) están sintácticamente asociados

Definiciones (4): traducción válida

Una **traducción válida** es un mapeo de una tokenización completa de q en elementos de E que cumple 3 condiciones:

1. Cada token se corresponde con un único elemento de E.
2. Cada token de atributo se relaciona con un único token de valor, cumpliendo que:
   - el atributo de la base de datos que corresponde al token de atributo y el valor de la base de datos que corresponde al token de valor son compatibles
   - ambos tokens están sintácticamente asociados
3. Cada token de relación está relacionado a un token de atributo o bien a un token de valor, cumpliendo las siguientes condiciones:
   - la relación de la base de datos que corresponde al token de relación y el elemento de la base de datos que corresponde al token de atributo o token de valor son compatibles
   - ambos tokens (token de relación - token de atributo o bien token de relación - token de valor) están sintácticamente asociados

Definiciones (4): traducción válida

Una **traducción válida** es un mapeo de una tokenización completa de q en elementos de E que cumple 3 condiciones:

1. Cada token se corresponde con un único elemento de E.
2. Cada token de atributo se relaciona con un único token de valor, cumpliendo que:
   - el atributo de la base de datos que corresponde al token de atributo y el valor de la base de datos que corresponde al token de valor son compatibles
   - ambos tokens están sintácticamente asociados
3. Cada token de relación está relacionado a un token de atributo o bien a un token de valor, cumpliendo las siguientes condiciones:
   - la relación de la base de datos que corresponde al token de relación y el elemento de la base de datos que corresponde al token de atributo o token de valor son compatibles
   - ambos tokens (token de relación - token de atributo o bien token de relación - token de valor) están sintácticamente asociados

Definiciones (5): Semánticamente tratable

Si una pregunta tiene solo una q-word y al menos una traducción válida entonces es **semánticamente tratable**

Y cada traducción válida de q determina una posible consulta SQL:

SELECT	Elementos apareados con qwords
WHERE	Pares de atributos y valores generados por el Matcher
FROM	Todas las relaciones mencionadas

Precise

El sistema utilizado por Popescu et al.

- Dada una pregunta q, determina si es semánticamente tratable y si lo es, genera una consulta SQL correspondiente. Si no, la rechaza pidiendo una reformulación.

Lexicon

- Sinónimos lematizados de todas las relaciones, atributos y valores
  - Wordnet
- Determina qué términos son comprensibles de una pregunta
- Las preguntas tratables tienen solamente marcadores sintácticos + palabras del lexicon.

Matcher

- La pieza clave.
- Relaciona cada valor con un único atributo, implícito o explícito usando un algoritmo de max flow.

Precise / módulos

- Lexicon: encargado de generar, para cada elemento de la base de datos, el conjunto de tokens sinónimos que se utilizará para verificar correspondencia.
- Tokenizer: encargado de generar todas las tokenizaciones completas de la pregunta y, para cada token, consultar al lexicon y retornar la lista de elementos de la base de datos que le corresponden.
- Matcher: encargado de verificar que las tokenizaciones completas y los elementos correspondientes generados por el tokenizer cumplan las condiciones 1 a 3 utilizando el modelo de grafos y el algoritmo max-flow recién ilustrado.
- Parser plug-in: el módulo computa la función de asociación sintáctica, basándose en el parse tree de Charniak.
- Query generator: dado el conjunto de elementos de la base de datos traducido de la pregunta, genera una query SQL.
- Equivalence Checker: verifica si diferentes queries son la misma formulada de diferentes maneras.

Ejemplo

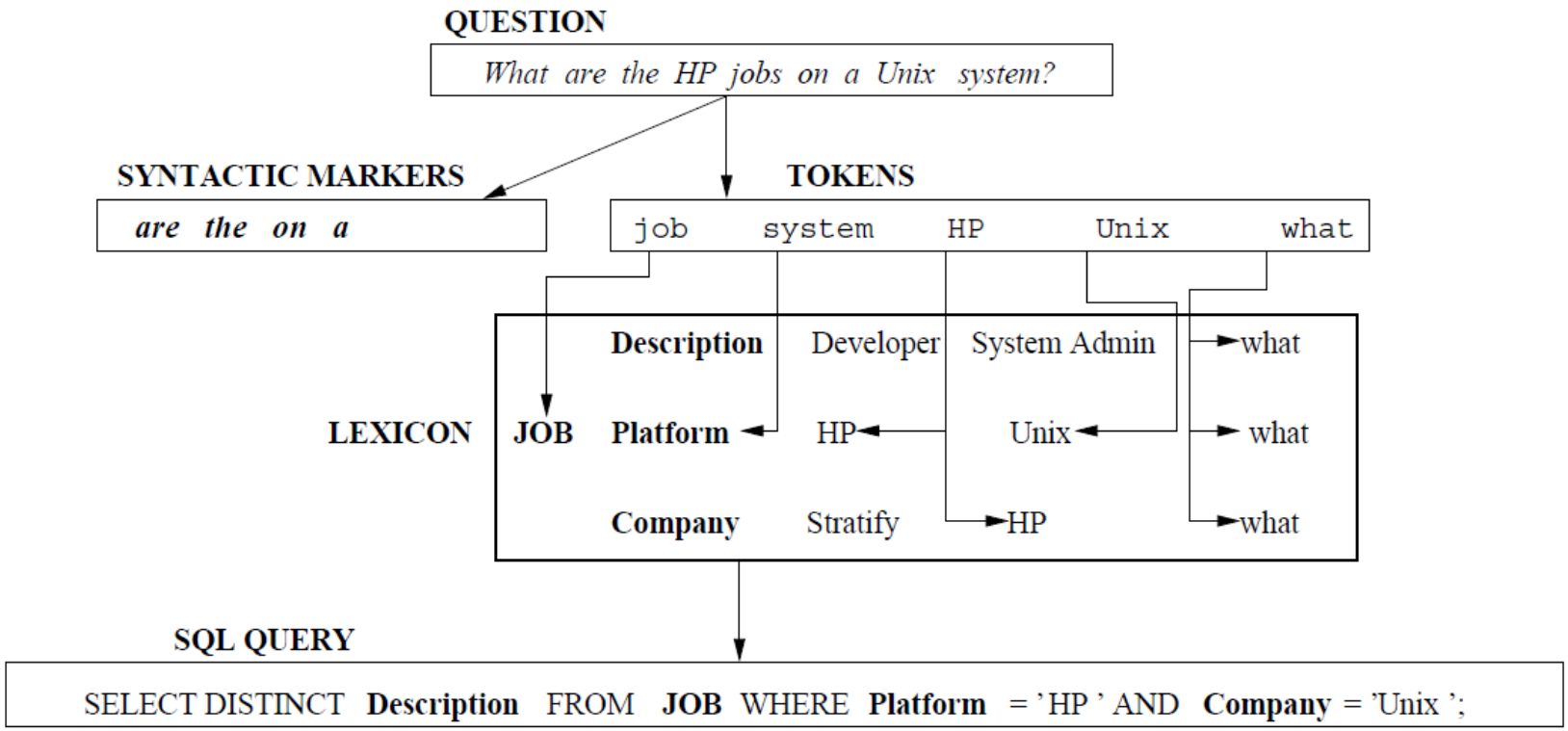


Figura: La traducción de la pregunta "What are the HP jobs on a Unix system?" a una consulta SQL

Ejemplo

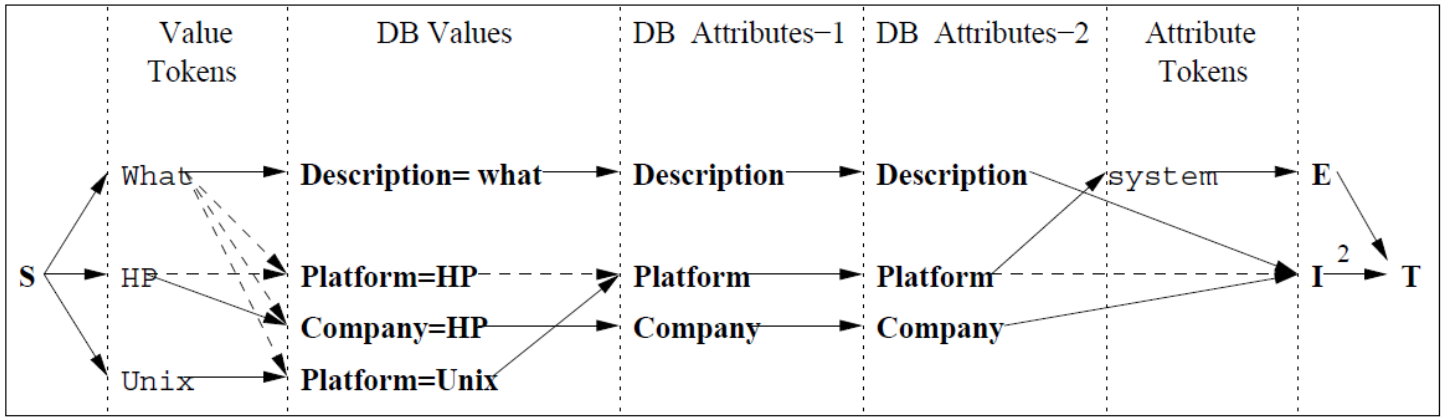
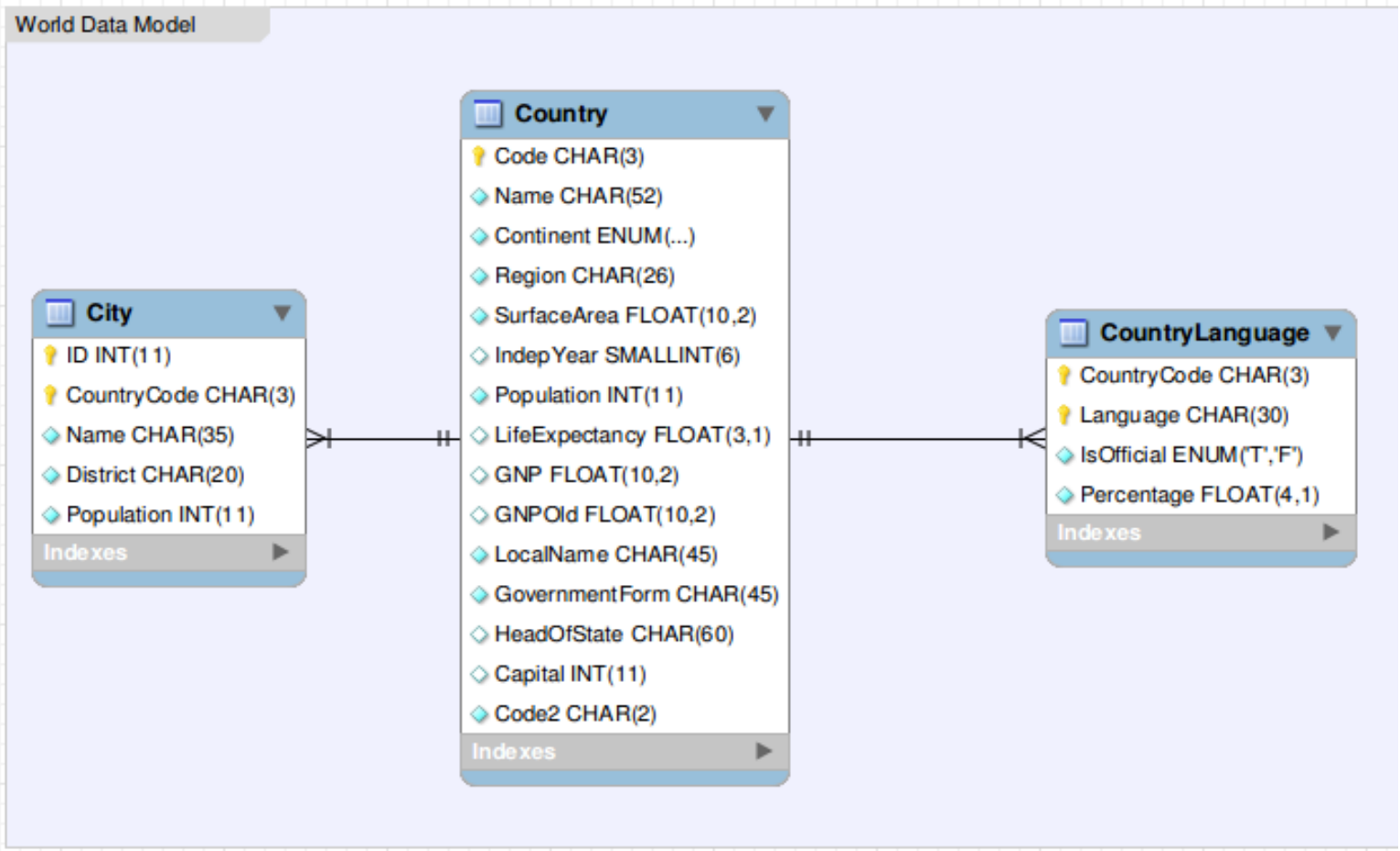


Figura: El grafo de atributos y valores creado por Precise para la pregunta “What are the HP jobs on a Unix system?”

index.tex

Base de datos: World (Country, City y CountryLanguage)



Lexicón

El lexicón es el módulo encargado de generar un conjunto de tokens para cada elemento de la base de datos. Una vez construido este conjunto, las responsabilidades del módulo son las siguientes:

- Dado un lema, devolver el conjunto de tokens que lo contienen (*getTokens*()).
- Dado un token, devolver el conjunto de elementos de la base de datos que le *corresponden* (*getMatchingElements*()).
- Wordnet. TokenAugmenter. Polisemia.

TokenAugmenter

Elemento original	Sinónimos
head of state	president, leader, emperor, king
region	location
surface area	size, total size, square kilometers, km2
independence year	independent, independency

Cuadro: Sinónimos introducidos por el Token Augmenter

Compatibilidad de Qwords

Qword	Atributos relacionados
What	**Name**, District, Population, Code, Continent, SurfaceArea, LifeExpectancy, GNP, LocalName, GovernmentForm, Capital, IsOfficial, Percentage, Region
Which	Los mismos que para what
Where	**Region**, Continent, Capital, District
Who	**HeadOfState**
When	**IndependenceYear**

Cuadro: Atributos compatibles con cada Qword

Tokenizer

1. Separar la pregunta en palabras, eliminar puntuaciones y pasar a lower case.
2. Lematizar las palabras. Para esto usamos Freeling
3. Eliminar marcadores sintácticos.
4. Para cada lema, obtener todos los tokens que lo contienen del Lexicon (método getTokens).
5. Para cada token potencial (resultado del paso anterior) verificar que todos sus lemas estén presentes en el conjunto de lemas de la pregunta original.
6. Generar el conjunto de partes de todos los tokens hasta aquí obtenidos. Probando con cada elemento del conjunto de partes en lugar de utilizar solamente el conjunto original podemos obtener subconjuntos que cumplan también la condiciones requeridas para considerarse una tokenización completa (evaluados en 7).
7. Para cada uno de estos subconjuntos, verificar 1) que sus tokens cubran completamente los lemas significativos de la pregunta original y 2) que no haya lemas repetidos entre los tokens.

Matcher

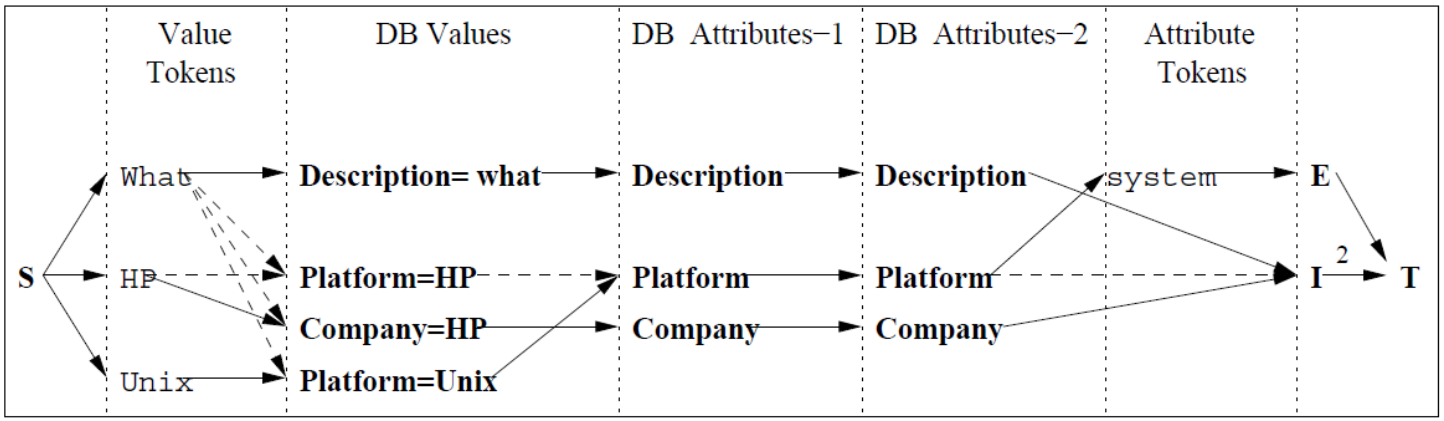


Figura: Matcher para “What are the HP jobs on a Unix system?”

Matcher

- Input = tokenizaciones completas generadas por el Tokenizer,
- Construye el grafo que expusimos en la sección NADA
- Algoritmo de max flow.
- Las aristas implicadas en el flujo máximo posible asocian
  - 1) los tokens de valor y de atributo y los correspondientes elementos (valores y atributos, respectivamente) de la base de datos y
  - 2) pares de valores y atributos entre sí.
- Otras soluciones posibles. Si el flujo es igual a la cantidad de tokens de valor en la pregunta es potencialmente una traducción válida. Retiramos ordenadamente las aristas del grafo que ocurren entre la columna 2 y 3 (tokens de valor y valores de la db).

Matcher: condiciones

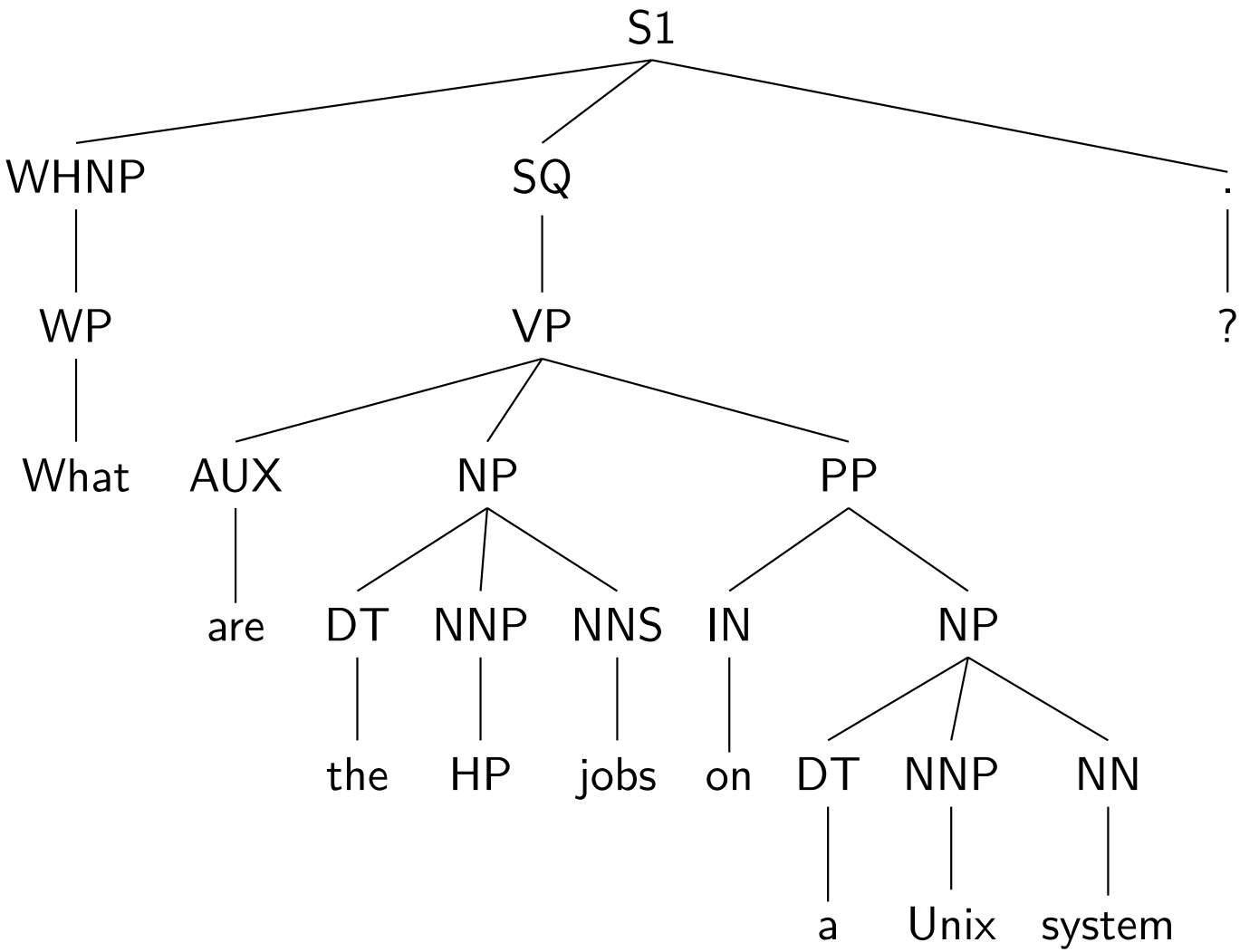
Finalmente, verificamos cuales de las soluciones con máximo flujo cumplen las condiciones requeridas para ser una traducción válida según enunciamos en **??**:

1. Todos los tokens de la tokenización tienen un único elemento de la base de datos asociado y no hay elementos de la base de datos repetidos. (Mapping.meetsConditionOne())
2. Cada token de atributo se relaciona con un único token de valor respetando que: (Mapping.meetsConditionTwo())
   1. el atributo relacionado con el token de atributo y el valor relacionado con el token de valor son compatibles (esta condición está garantizada por el max-flow mismo)
   2. ambos tokens están sintácticamente asociados
3. Cada token de relación está relacionado a un token de atributo o bien a un token de valor, cumpliendo las siguientes condiciones: (Mapping.meetsConditionThree())
   1. la relación de la base de datos que corresponde al token de relación y el elemento de la base de datos que corresponde al token de atributo o token de valor son compatibles
   2. ambos tokens (token de relación - token de atributo o bien

Charniak: chequeo sintáctico

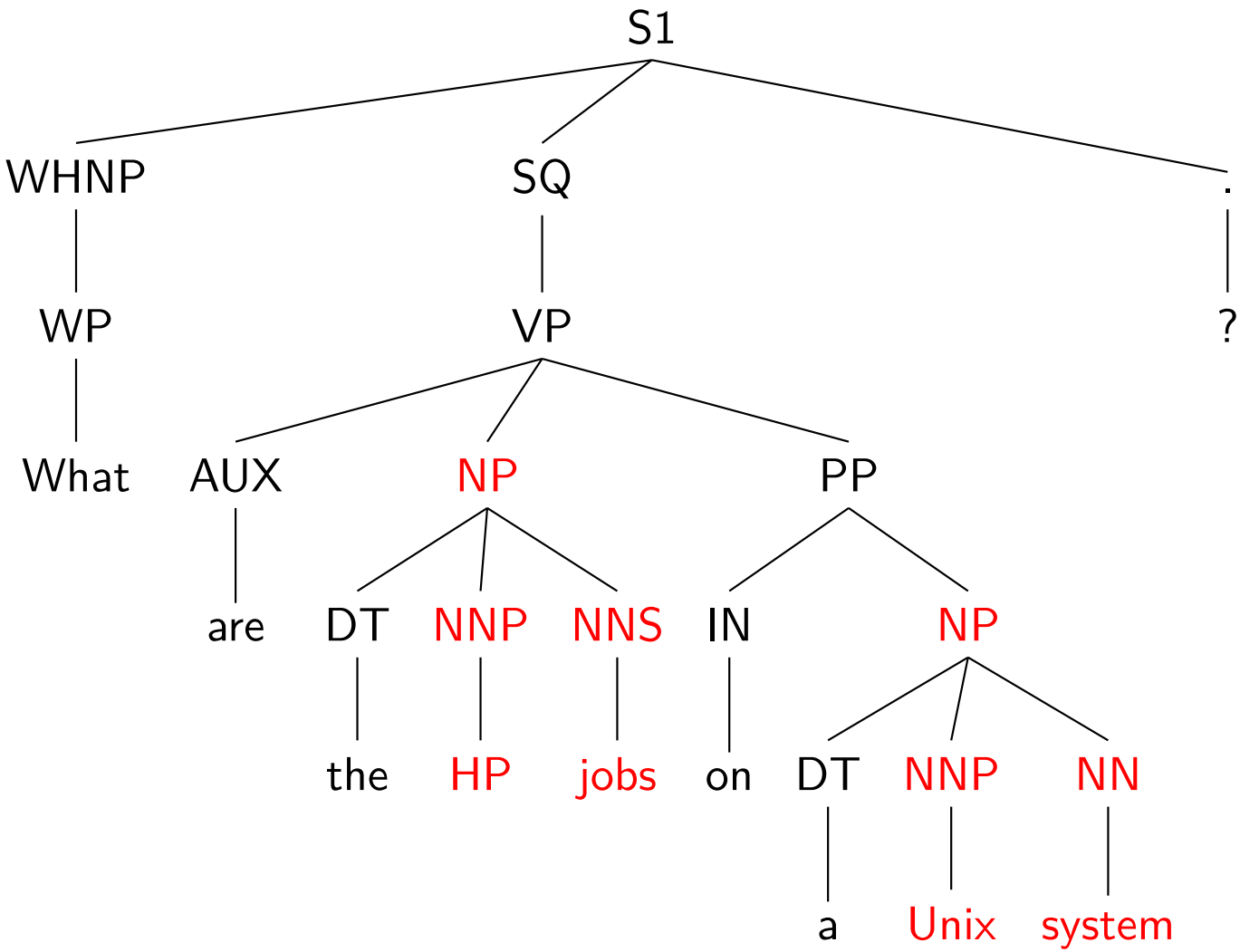
- No especificado. Experimental. Sin virtuosismos
- Vinculan hoja sustantivo, adjetivo o adverbio → (JJ, JJR, JJS, NN, NNS, NNP, NNPS, RB, RBR, RBS, CD)
- Hojas sustantivo →)NN, NNS, NNP, NNPS)
- Qwords → (WDT, WP, WP$, WRB)

Charniak



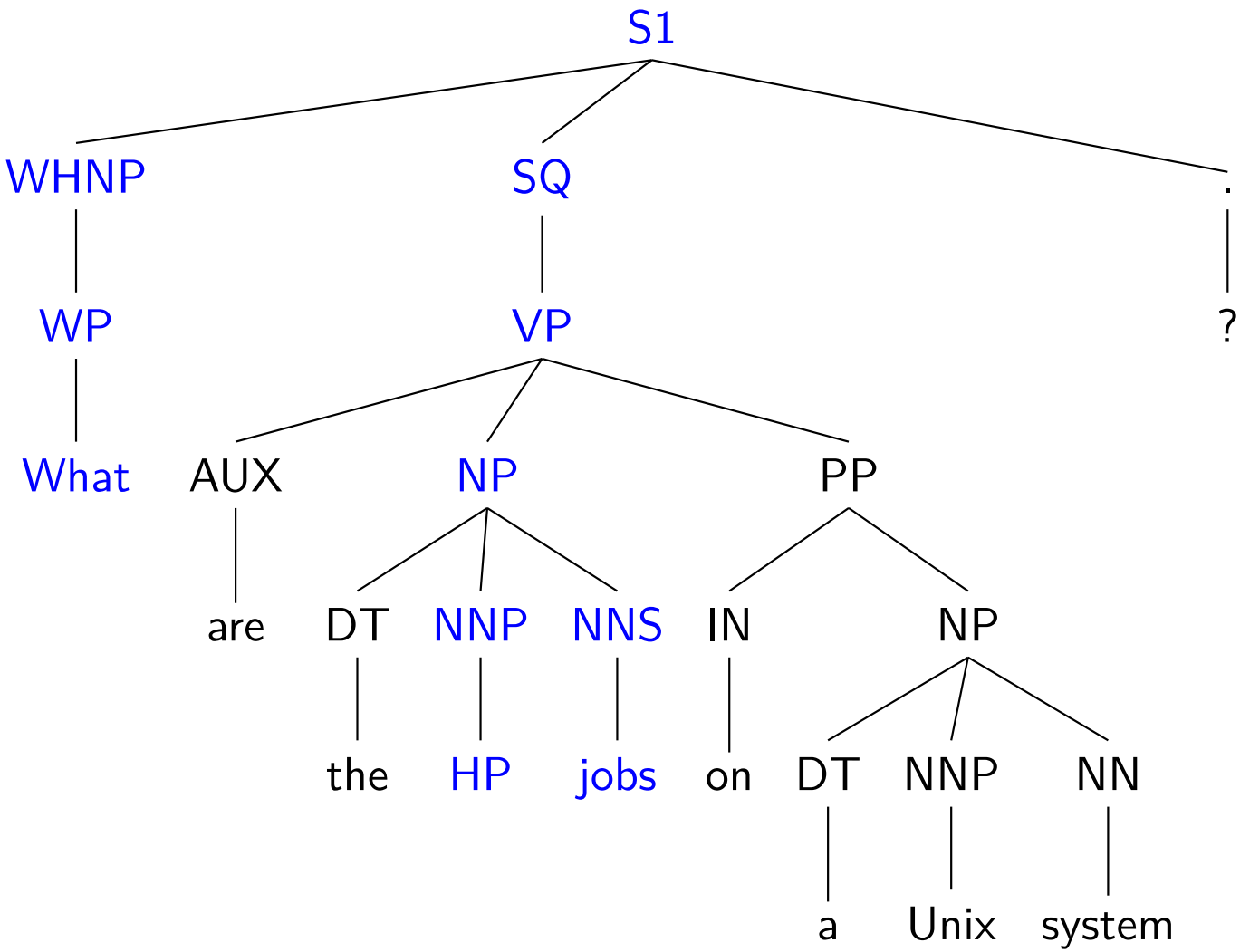
"What are the HP jobs on a Unix system?"

Charniak



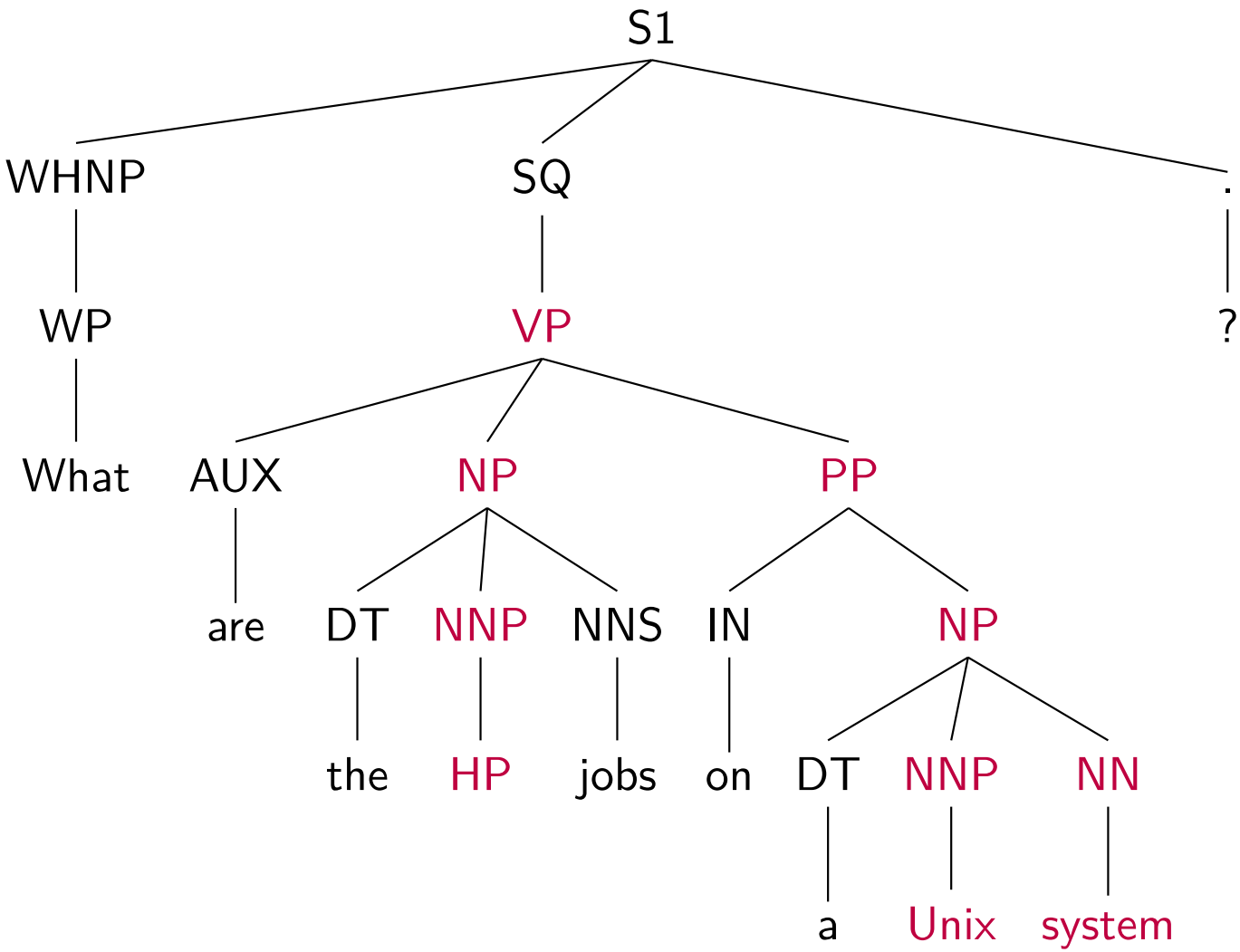
Regla 1 - Sustantivos “hermanos”

Charniak



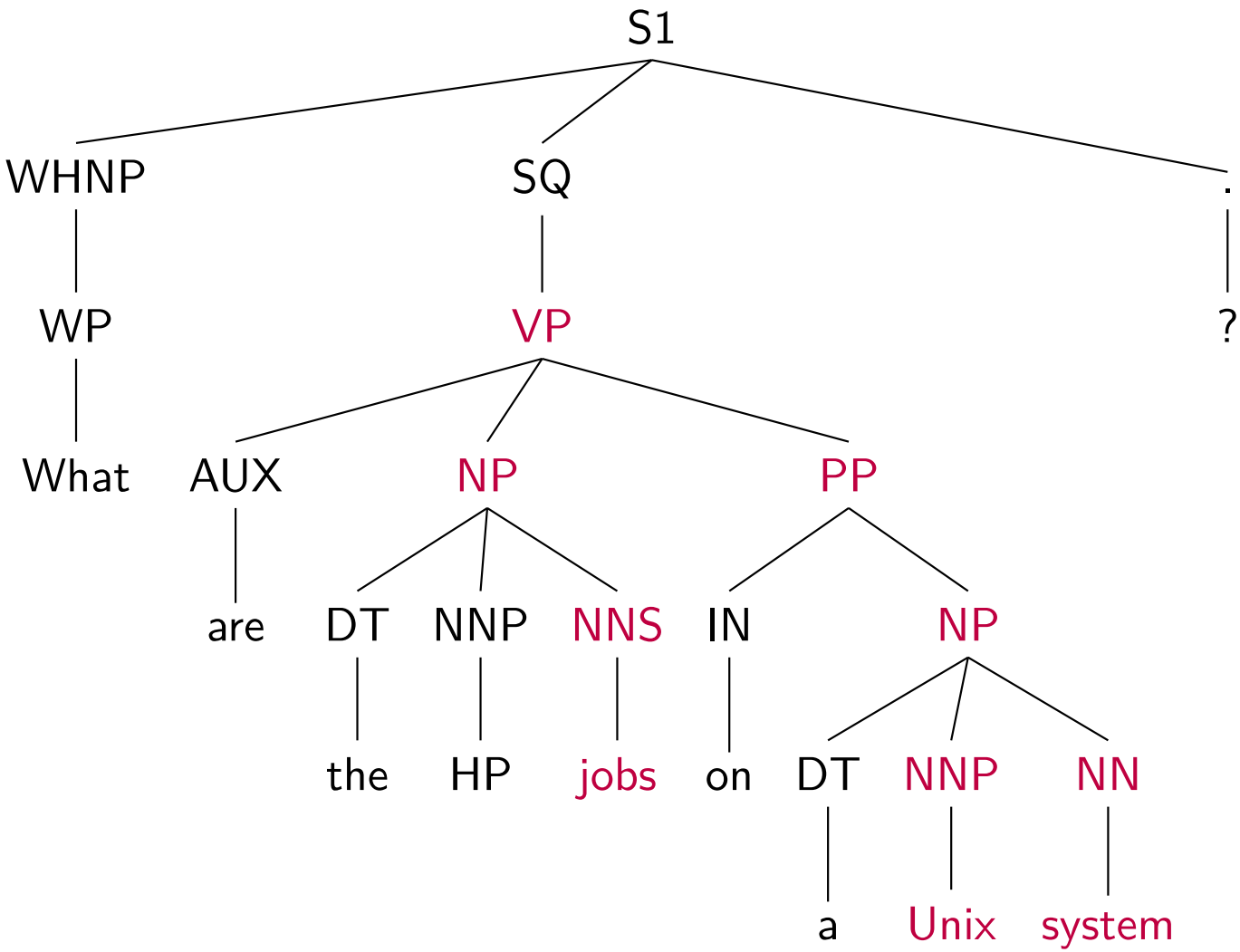
Regla 2: qwords a sustantivos

Charniak



Regla 3: sintagma nominal a sintagma preposicional

Charniak



Regla 3: sintagma nominal a sintagma preposicional (otra)

Matcher, filtro, queries

No usar charniak. Mejora la performance. Empeora filtro. Mas desambiguaciones mas tontas.
Finalmente, todos los resultados de max-flow que cumplen con las condiciones de 1 a 4 son traducciones válidas, que pasan al MappingFilter, que realiza ciertos filtrados que describiremos, de nuevo, en un título aparte. Resultado e Mappings (una estructura que contiene 1) una tokenización completa de la pregunta original y 2) un mapeo válido entre cada token de la misma en un elemento de la base de datos). Cada mapeo es una traducción válida de la pregunta. Si existe solo uno, entonces este mapeo se traducirá en una query que generará el resultado. Si no, corresponde al MappingFilter realizar filtrados inteligentes de las múltiples soluciones y, en caso de que continuen existiendo múltiples soluciones, entonces se consultará al usuario qué quiso preguntar. Por otro lado, si no fue posible generar ninguna traducción válida, se retornará al usuario sin respuesta, pidiéndole que vuelva a escribir su pregunta de otro modo.

Filtros, Generación de queries

MappingFilter

- Elimina repetidas (según su query SQL).
- Regla 1 - "semi duplicados": atributos similares de la DB[1]
- Regla 2 y 3 - Fundamento. si una traducción válida tiene su qword asociado a un atributo *implícito*, también estará asociado con todos los atributos que pueda estarlo (ver tabla table:atributos-qwords para las compatibilidades definidas).[2]
- Regla 2 - Preferir, entre los atributos implícitos, aquellos cuya relación está mencionada en la pregunta (si hubiera alguna).
- Regla 3 - Una vez especificada la relación, priorizar el atributo implicito preferido en negrita en la tabla table:atributos-qwords.
- Si después de MappingFilter hay más de una traducción, vuelve al usuario a desambiguar

QueryGenerator

SELECT	Elementos apareados con qwords
WHERE	Pares de atributos y valores generados por el Matcher
FROM	Todas las relaciones mencionadas

[1] en concreto, una query que consulta por Country.Name Country.LocalName

[2] Esto implica que, por ejemplo, para cualquier pregunta cuya qword sea 'what' y no tenga su atributo asociado explícito, habrá 16 diferentes traducciones válidas.

Ejemplos

- FILL ME
- Who is the head of state of Zimbabwe?
- What caribbean countries are also considered north american?

Conclusiones y limitaciones

- FILL ME

Ejemplo 1: “Who is the head of state of Zimbabwe?”

Ejemplo 1:

“Who is the head of state of Zimbabwe?”

Ejemplo 1: Tokenizer

1. Separar en palabras, eliminar puntuaciones y pasar a lower case:

"who is the head of state of zimbabwe?"

Ejemplo 1: Tokenizer

1. Separar en palabras, eliminar puntuaciones y pasar a lower case:

$$\{\text{who}, \text{is}, \text{the}, \text{head}, \text{of}, \text{state}, \text{of}, \text{zimbabwe}\}$$

Ejemplo 1: Tokenizer (2)

2. Lematizar las palabras, eliminar marcadores sintácticos:

{who, ~~is~~, ~~the~~, head, ~~of~~, state, ~~of~~, zimbabwe}

Ejemplo 1: Tokenizer (2)

2. Lematizar las palabras, eliminar marcadores sintácticos:

$\{\text{who, head, state, zimbabwe}\}$

Ejemplo 1: Tokenizer - tokens para who

3. Obtener tokens para cada lema

$$getTokens(who) \rightarrow \{\text{'who'}\}$$

Ejemplo 1: Tokenizer - tokens para head

3. Obtener tokens para cada lema

$getTokens(head) \rightarrow$ { 'head country', 'head teacher body politic', 'head body politic', 'head teacher land', 'read / write head dos', 'heading provincial', 'heading state', 'heading commonwealth', 'read / write head body politic', 'head word land', 'head state of matter', 'read / write head provincial', 'read / write head state department', 'head province', 'heading res publica', 'read / write head nation', (51 más)...}

Ejemplo 1: Tokenizer - tokens para state

3. Obtener tokens para cada lema

$getTokens(state) \rightarrow$ {'heart eastern united states', 'centre eastern united states', 'central eastern united states', 'centrical eastern united states', 'midsection eastern united states', 'midriff eastern united states', 'centric eastern united states', 'center eastern united states', 'north western united states', (702 más)...}

Ejemplo 1: Tokenizer - tokens para Zimbabwe

3. Obtener tokens para cada lema

$getTokens(zimbabwe) \rightarrow$ {'zimbabwe', 'republic of zimbabwe', 'capital of zimbabwe'}

Ejemplo 1: Tokenizer - intersección

4. Filtrar solo tokens cuyos lemas estén en la pregunta

$getTokens(zimbabwe) \rightarrow \{$'zimbabwe', ~~'republic of zimbabwe'~~, ~~'capital of zimbabwe'~~$\}$

Ejemplo 1: Tokenizer - intersección

4. Filtrar solo tokens cuyos lemas estén en la pregunta

$$getTokens(zimbabwe) \rightarrow \{\text{'zimbabwe'}\}$$

Ejemplo 1: Tokenizer - intersección

4. Filtrar solo tokens cuyos lemas estén en la pregunta

$$getTokens(zimbabwe) \rightarrow \{\text{'zimbabwe'}\}$$
$$getTokens(who) \rightarrow \{\text{'who'}\}$$
$$getTokens(head) \rightarrow \{\text{'head state','heading state'}\}$$
$$getTokens(state) \rightarrow \{\text{'state','head state','heading state'}\}$$

Ejemplo 1: "Who is the head of state of Zimbabwe?"

Generar el conjunto de partes de todos los tokens.

∅,
{'zimbabwe', 'state'},
{'zimbabwe', 'state', 'head state', 'who'},
{'head state', 'zimbabwe', 'who'},
{'heading state', 'zimbabwe', 'head state', 'who'},
...

Para cada uno de estas *tokenizaciones*, verificar:

- Que sus palabras cubran la pregunta original.
- Que no haya palabras repetidas entre los tokens.

Ejemplo 1: Tokenizaciones completas

$$q = \text{“Who is the head of state of Zimbabwe?”}$$

$$CompleteTokenizations(q) = \begin{cases} \text{who, head state, zimbabwe} \\ \\ \text{who, heading state, zimbabwe} \end{cases}$$

Ejemplo 1: Matcher - Tokens $\rightarrow$ Elementos

Obtener elementos de la DB para cada token

zimbabwe $\rightarrow$ *Zimbabwe* (Valor de Country.Name)

Ejemplo 1: Matcher - Tokens $\rightarrow$ Elementos

Obtener elementos de la DB para cada token

head state $\rightarrow$ *HeadOfState* (Atributo de Country)

Ejemplo 1: Matcher - Tokens → Elementos

Obtener elementos de la DB para cada token

heading state → *HeadOfState* (Atributo de Country)

Ejemplo 1: Matcher - Tokens $\rightarrow$ Elementos

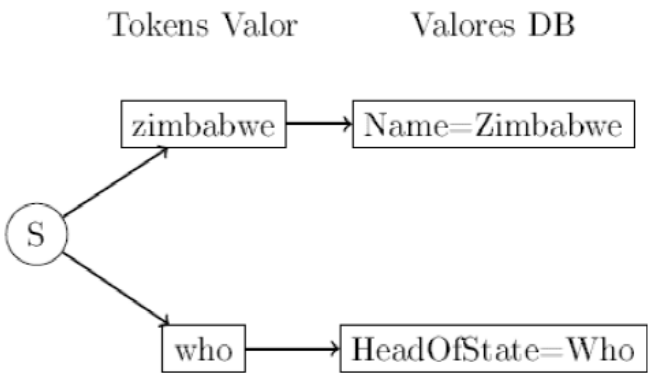
Obtener elementos de la DB para cada token

who $\rightarrow$ *Who* (Qword compatible con HeadOfState)

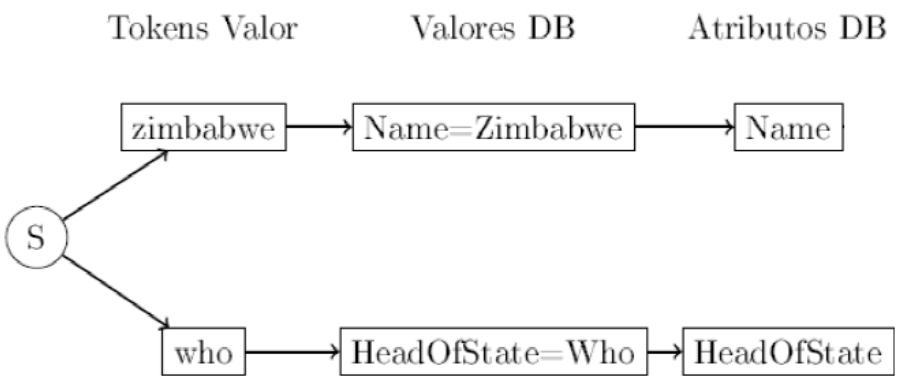
Ejemplo 1: Matcher - Grafo de atributos y valores

Ejemplo 1: Matcher - Grafo de atributos y valores

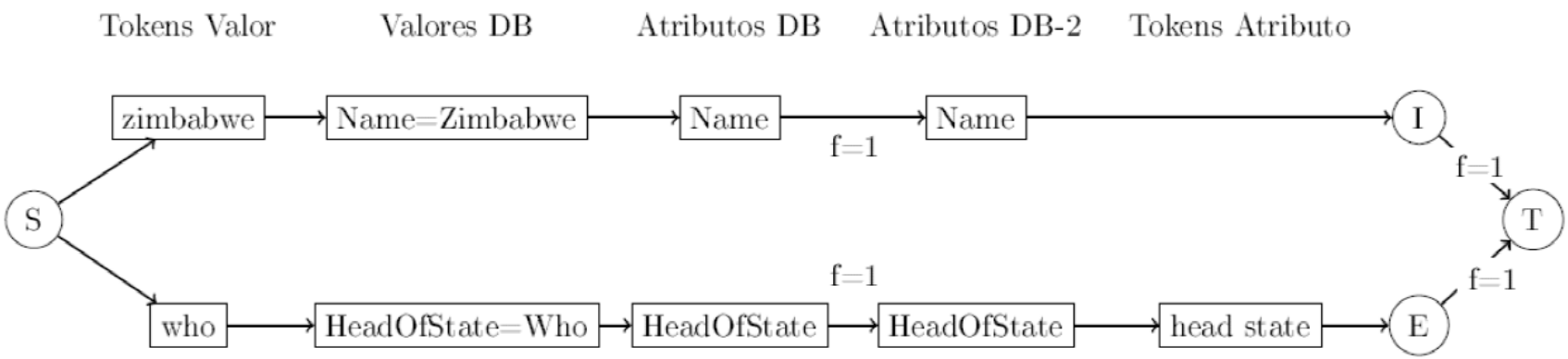
Ejemplo 1: Matcher - Grafo de atributos y valores



Ejemplo 1: Matcher - Grafo de atributos y valores



Ejemplo 1: Matcher - Grafo de atributos y valores



Ejemplo 1: Matcher - notas

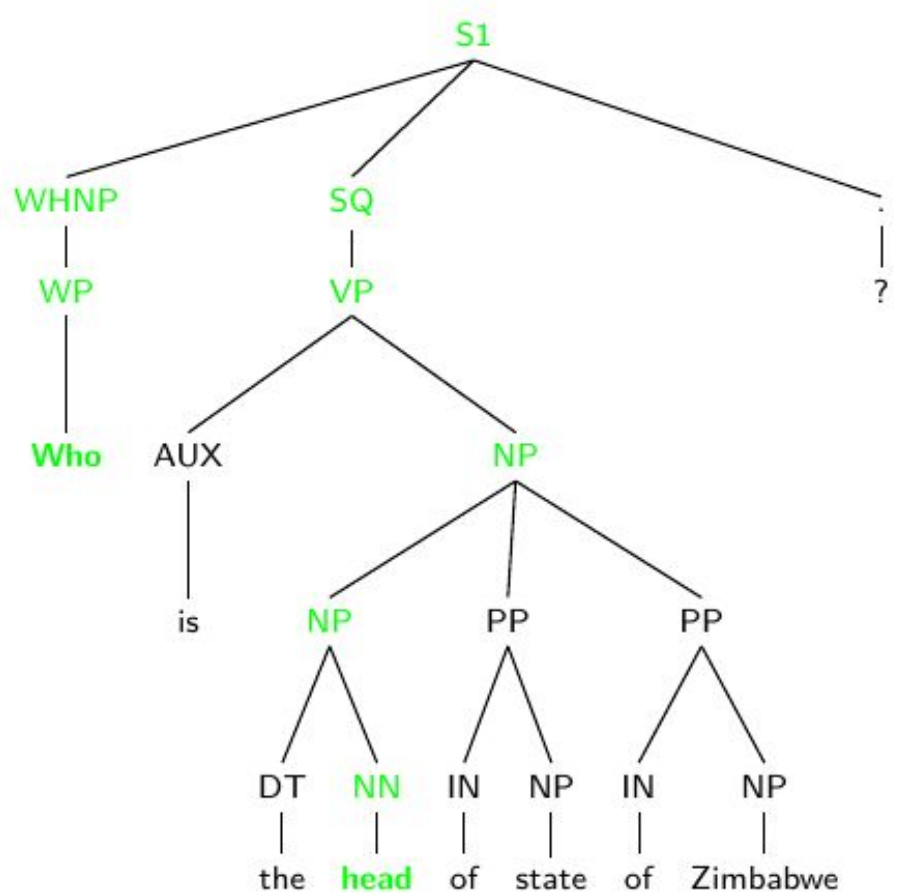
"Who ~~is~~ ~~the~~ head ~~of~~ state ~~of~~ Zimbabwe?"

- head state es el atributo Country.HeadOfState
- Who es un valor (especial) de Country.HeadOfState
- Zimbabwe es un valor de Country.Name (implícito)
- No desambigua nada (los tokens tiene un solo elemento asociado)

Los tokens 'Who' y 'Head state' deben estar sintácticamente asociados

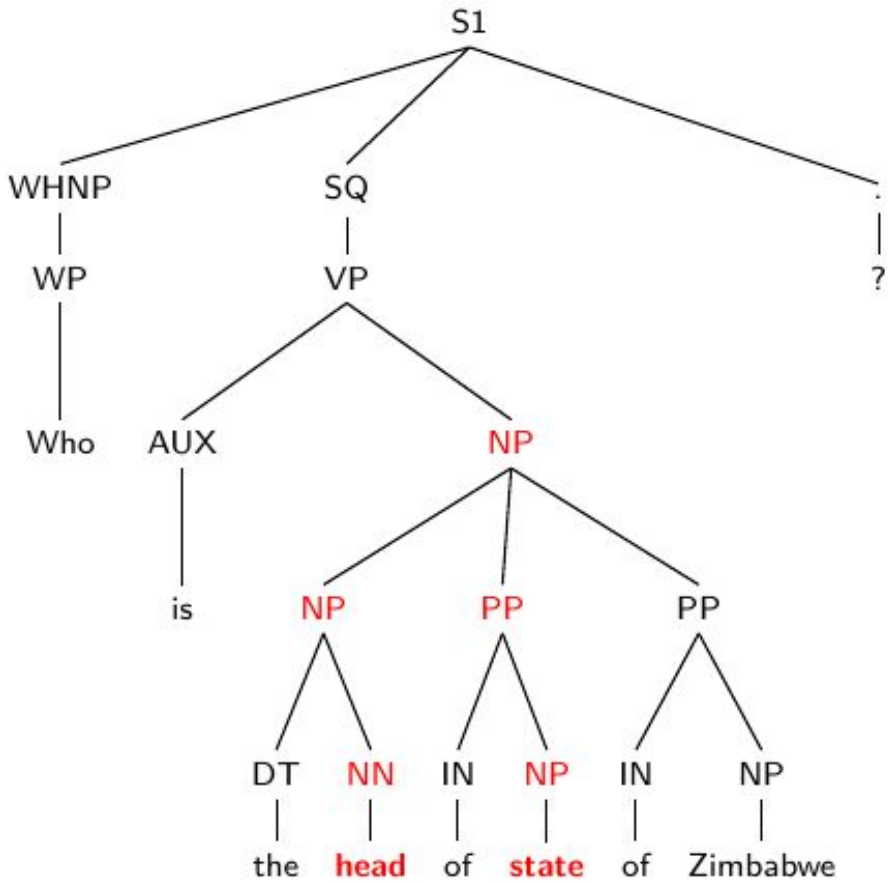
Ejemplo 1: Matcher - Asociación sintáctica - Regla 1

Regla 1: who → head



Ejemplo 1: Matcher - Asociación sintáctica - Regla 6

Regla 6: head → state



Ejemplo 1: Final

Who is the head of state of Zimbabwe?

Ejemplo 1: Final

Who ~~is~~ ~~the~~ head ~~of~~ state ~~of~~ Zimbabwe?

Ejemplo 1: Final

Who ~~is~~ ~~the~~ head ~~of~~ state ~~of~~ Zimbabwe?

Who → Valor de Country.HeadOfState
head state → Atributo Country.HeadOfState
Zimbabwe → Valor implícito de Country.Name

Ejemplo 1: Final

Who ~~is~~ ~~the~~ head ~~of~~ state ~~of~~ Zimbabwe?

SELECT HeadOfState
FROM Country
WHERE Country.Name= 'Zimbabwe'

"Robert G. Mugabe"

index.tex

index.tex

Introducción

- Sistema QA de dominio abierto con soporte multilingüe
- Basado en Qanus, adaptado para usar Freeling.
- Ejercicios de CLEF'07 monolingüe para español y portugués
- Wikipedia como base de conocimiento
- IR + NLP + Heurísticas
- Agregamos:
  - Heurísticas de generación de queries (de Lasso y de *topics*)
  - Heurística de ranking de pasajes
- Experimentamos y sacamos conclusiones.

index.tex

Marco teórico

Consenso general sobre la arquitectura: pipeline de al menos 3 componentes:

- Módulo de procesamiento de la pregunta
  - Submódulo principal: Question Classifier
  - Anotaciones de la pregunta (NER, POS, QC, Otros: focus, answer type)
  - Expansión/reformulación de la consulta (Heurística de Lampert, Moldovan)
- Módulo de procesamiento de documentos
  - Submódulo principal: índice invertido (estructuras de IR)
  - (Otros: división en parágrafos, filtrado grueso, re-ranking)
- Módulo de procesamiento de la respuesta
  - Valor agregado ppal de un sistema de QA sobre uno de IR
  - El menos estandarizado de los tres
  - Heurísticas puntuales basadas en casos.

Dominio de problemas, módulos

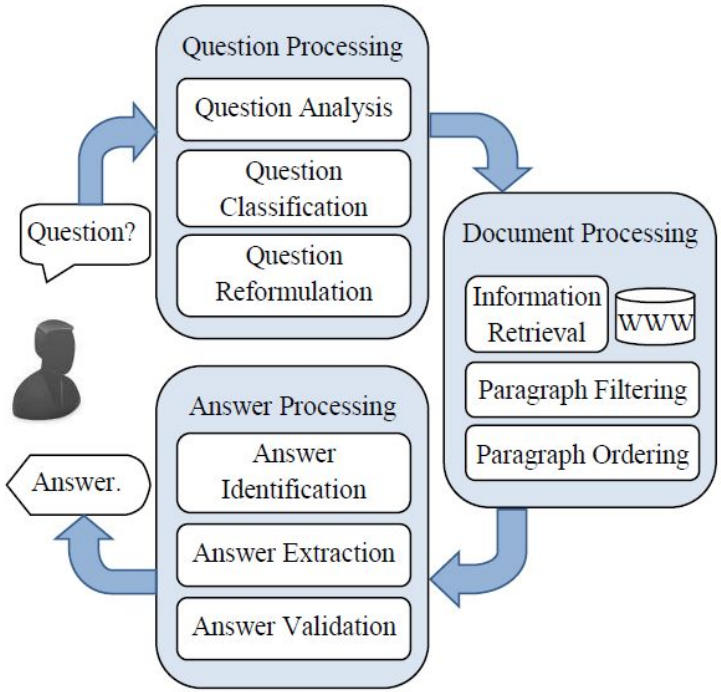


Figura: Módulos y submódulos de un sistema de QA

Dominio general

- Cualquier tema y tipo de pregunta
- Corpora de texto $\rightarrow$ datos no estructurados
- Question answering como sistema de pipeline
- IR + NLP + Heurísticas

Heurística de generación de queries [Lampert, 2004]

8 pasos

1. Palabras no stop words entre comillas
2. Entidades nombradas reconocidas
3. Construcciones nominales con sus adjetivos
4. Demás construcciones nominales
5. Sustantivos con sus adjetivos
6. Demás sustantivos
7. Verbos
8. El 'focus' de la pregunta

index.tex

Ejercicios

Cross-Language Evaluation Forum

- Principal organizador de conferencias de question answering multilingüe para idiomas europeos.
- La tarea principal de '07 presentaba una complejidad adecuada para el scope de esta tesis.

Ejercicios

- Subset de dos ejercicios de la tarea principal de question answering de '07.
- Corpora, inputs de prueba y resultados esperados para una experimentación sobre diferentes idiomas disponible.
- Wikipedia + newsletters como corpus
  - Nosotros trabajamos solo con las wikipedias
- 200 preguntas (factoids, definition o list)
- Respuesta exacta. Topics + clusters + co-referencia.
- Preguntas sin respuesta
- Tomamos ejercicios monoligüe es-es y pt-pt (español-español y portugués-portugués)

Tareas activadas en Clef '07

Table 1: Tasks activated in 2007 (in green)

SOURCE LANGUAGES (questions) \ TARGET LANGUAGES (corpus and answers)	BG	DE	EN	ES	FR	IT	NL	PT	RO
BG	■		■	■					
DE		■	■	■					
EN	■	■		■	■	■	■	■	■
ES			■	■	■			■	
FR			■	■	■				
IN			■						
IT		■	■	■		■			
NL			■	■			■		
PT		■	■	■	■			■	
RO			■	■					■

Figura: Tareas activadas en la competencia Clef '07

Preguntas

Subset	Idioma	Factoid	Definition	List	Total
Todo	es	158	32	10	200
	pt	159	31	10	200
Wiki	es	122	24	9	163
	pt	104	18	8	130

Cuadro: Totales por tipo de pregunta

Ejemplo de grupo con tema y correferencia (primer cluster de preguntas para es-es):

- ¿En qué colegio estudia Harry Potter?
- ¿Cuál es el lema del colegio?
- ¿En qué casas está dividido?
- ¿Quién es el director del colegio?

Qanus

Question-Answering @ National University of Singapore

- Un framework de question answering basado en information retrieval
- QA-sys: un sistema de QA funcional simple construido sobre este framework.
- Actualizado por última vez en noviembre de 2012, código abierto
- Herramientas de NLP de Stanford (POS, NER y QC, para inglés) y Lucene.
- Estructura de Pipeline. Orientado para TREC 07 (Aquaint)

Qanus

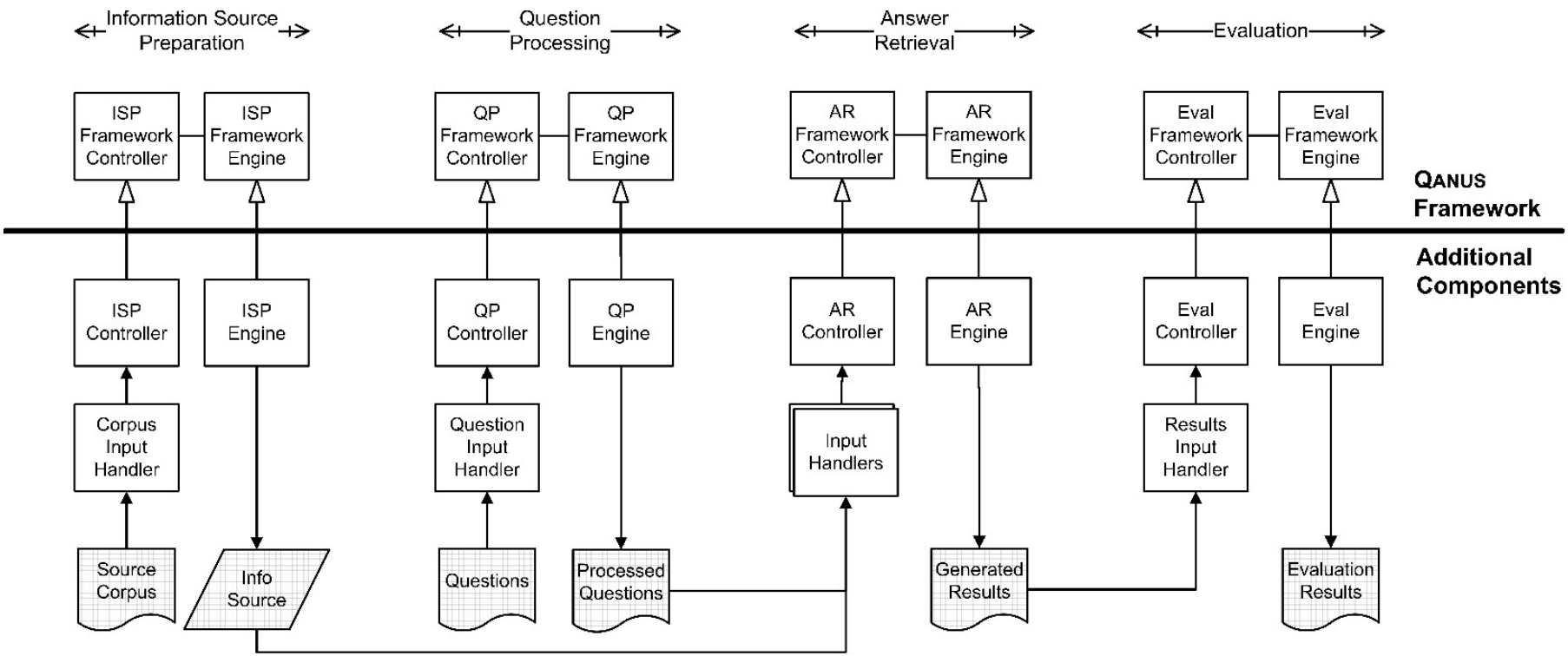


Figura: El framework Quanus y la implementación QA-sys

QA-Sys - Qanus @ Trec 07

Run Tag	Accuracy
LymbaPA07	0.706
LCCFerret	0.494
lsv2007c	0.289
UofL	0.258
QASCU1	0.256
FDUQAT16A	0.236
pronto07run3	0.222
ILQUA1	0.222
Ephyra3	0.208
QUANTA	0.206

Figura: Resultados de Trec 07

TREC 07

- Monolingue, Inglés. Competencia importante
- Una respuesta por pregunta
- 200 preguntas en total
- Evaluada por humanos
- *Precisión*: respuestas exactas sobre respuestas totales
- LymbaPA $\rightarrow$ 0.706 (70 %)
- QA-Sys $\rightarrow$ 0.119 (12 %)

Baseline: QA-Sys

1. Preparación de la fuente de información: Preprocesa XML de Aquaint de manera offline en un índice Lucene.

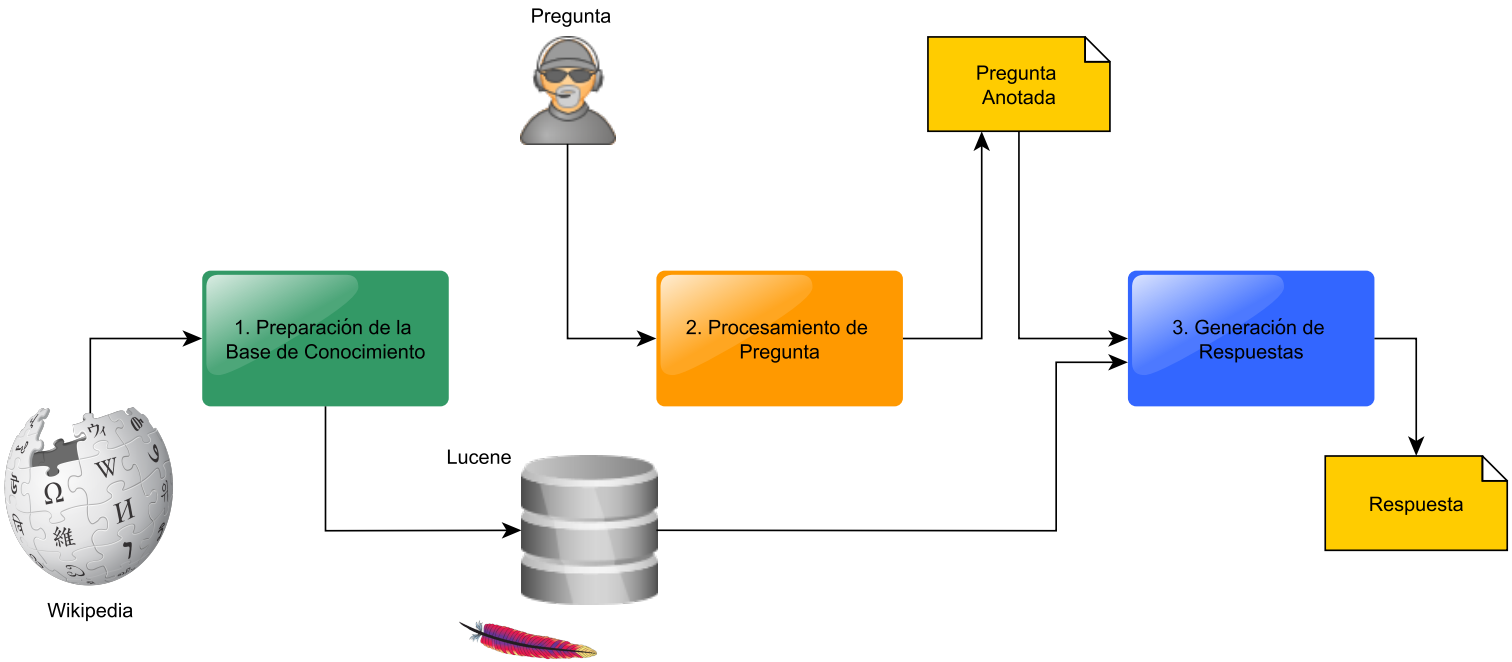
2. Análisis de la pregunta:
Anota la pregunta con POS, NER y QC.

3. Generación de respuestas: Busca la pregunta en el índice, separa los documentos en pasajes y los rankea con métricas heurísticas y luego, dependiendo del tipo de QC, aplica heurísticas basadas en NER, POS y QC sobre los pasajes para extraer una respuesta.

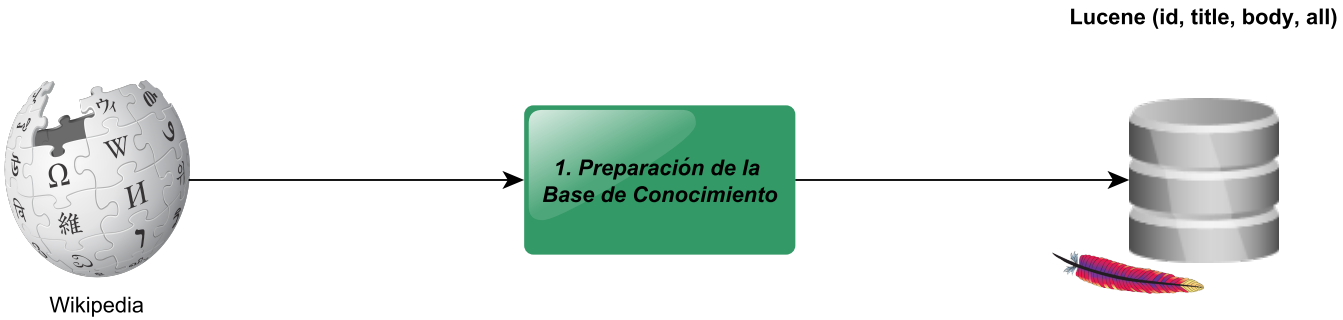
Baseline: ML QA-Sys

- Soporte multi idioma no simultaneo.
  - Soporta varios idiomas, pero de a uno a la vez.
- Indice invertido de Wikipedia
- Reemplazamos las herramientas NLP por Freeling y sacamos reglas dependiente al idioma.

Pipeline


Pregunta
Pregunta Anotada
1. Preparación de la Base de Conocimiento
2. Procesamiento de Pregunta
3. Generación de Respuestas
Lucene
Wikipedia
Respuesta

1. Preparación de la base de conocimiento



Wikipedia	# Entradas	Nulas	Redirects	Filtradas	**Válidas**
simple-06	18273	22	3452	5241	9558
simple-13	180067	8	35600	41902	102557
es-06	**233750**	52	62805	34947	**135946**
pt-07	**498039**	80	210983	43390	**243586**

Cuadro: Wikipedias: cantidad de entradas válidas e inválidas

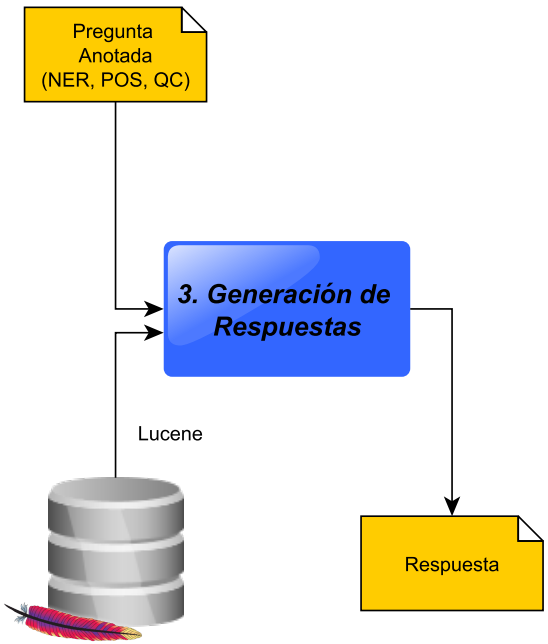
2. Procesamiento de la pregunta

- NER → Freeling
- POS → Freeling Explicar que es quizas con graficos?
- QC → No hay (tesista?)
  - Aprovechando que estaban activas las tareas en-es y en-pt
  - Hack (truchada) → pasamos QC sobre preguntas en inglés

Clase	# ES	# PT
HUM	62	53
NUM	53	52
ENTY	34	28
DESC	25	30
LOC	24	36
ABBR	2	1
Total	200	200

Cuadro: Clases de las preguntas para ES y PT, Stanford Classifier (Li & Roth, Stanford)

3. Generación de Respuestas



El algoritmo tiene los siguientes pasos:

1. Filtrado de preguntas
2. Generación de Queries
3. Information Retrieval
4. Extracción y rankeo de oraciones
5. POS tagging de las mejores oraciones
6. Heurísticas de AR basadas en el QC

3. Generación de Respuestas (2)

3.1 Filtrado de preguntas

- Solo factoids (no definition, no list, no nil), solo de wikipedia
- Total: 122 es, 104 pt

3.2 Generación de Queries

- Descartamos caso ("Who (is—was) (the) NN ... NN?")
- 2. Caso baseline
  - Espera 'target'(TREC) / tópico (Clef) y pregunta.
  - Query:
    - Palabras no stop-words de *target*
    - Sustantivos, verbos y adjetivos de la pregunta.

3. Generación de Respuestas (IR)

3.3, 3.4, 3.5 - IR + Extracción y rankeo de oraciones

- Documentos divididos en oraciones (no heredan rank)
- Peso ponderado por scorers entre 0 y 1:
- $rank = 0{,}9 \times covr + 0{,}05 \times freq + 0{,}05 \times prox$;
- Primeras K oraciones pos-taggeadas, el resto se desecha

Scorers: Freq, Cov, Prox

Freq & Cov: Ocurrencia de la query en la respuesta candidata

- Frecuencia: tokens de la query que ocurren en la oración sobre tokens de la oración
- Cubrimiento: tokens de que query que ocurren en la oración sobre tokens de la query

Prox: distancia de dos strings en un tercero

- s_1 =“Argentina es un país americano” y s_2 =“independizado en 1810”
- t =“Argentina es un país americano, originalmente una colonia española, independizado en 1810”
- $prox(s_1, s_2, t) = fracdist(s_1, s_2)len(t) = frac712 = 0{,}58$,
  - $dist(s_1, s_2)$ = cantidad de strings entre ‘un’ y ‘en’ (tokens intermedios de $s_{1,2}$)
  - $len(t)$ = cantidad de strings del texto
  - 1 denota que los dos string están cercanos uno al otro en el tercer string.

Scorers: Span

Span: similar a Prox, pero sobre un string y sus tokens

- $s = \{token_1, token_2, ..., token_n\}$ y t un texto
- Sea r la cantidad de tokens de s que aparecen en t
- Sea d la distancia de los tokens de s más distantes en t
- Entonces, Span = $fraqrd$
- Un score cercano a 1 significa que los términos del string buscado están cerca en el string en el que se buscan.
- Ejemplo (match es una X): ..... X ..... X ..... X ....a ..... b ..... c ..... $Span$ = #total de tokens encontrados / $|c - a|$.

3.6 Heurísticas de AR

3.6 Heurísticas de AR

- Basadas en Question Type
- Re ranking
- Extracción especifica

Casos

- ABBR:exp
- ABBR:abb $\rightarrow \emptyset$
- HUM:gr y ENTY:cremat
- HUM:ind
- HUM general
- LOC general
- NUM:date
- NUM:period
- NUM:count
- NUM general
- ENTY general.

Casos - ABBR:exp

Caso ABBR:exp - expansión de una abreviación

- Se extraen las entidades nombradas de tipo *Organización* de la pregunta.
- Se generan regex de expansión de esas abreviaciones.
  - UBA $\rightarrow$
    $[U][A-Za-z0-9][B][A-Za-z0-9][A][A-Za-z0-9]$.
- Busca este patrón en las 40 oraciones, devuelve el primer match.

Casos - HUM:gr y ENTY:cremat

Caso HUM:gr y ENTY:cremat - grupos humanos (compañías, organizaciones) y objetos humanos (libros, inventos, discos, etc)

- Respuestas candidatas $\rightarrow$ extraer nombres propios (proper nouns, NNPs) de todas las oraciones rankeadas .
  - Nombres propios consecutivos se agrupan (Tiger Woods)/NNP
- ~~Filtro candidatas que contengan palabras en diccionario.~~ [3]
- Re ranking de las candidatas según scores:
  - $Total = 0{,}55 * Coverage + 0{,}2 * Sentence + 0{,}1 * Proximity + 0{,}15 * RepeatedTerm$
  - *Coverage*: cubrimiento del target en la oración de soporte
  - *Proximity*: distancia entre el target y la respuesta candidata en la oración
  - *Sentence*: posición de la oración en las 40 seleccionadas
  - *RepeatedTerm*: penalización. negativa. si la respuesta candidata tiene tokens en común con el target

[1] En concreto: {Monday, Tuesday, Wednesday, Thursday, Friday, Saturday, Sunday}, {January, February, March, April, May, June, July, August, September, October, November, December}, y {us, uk, france, england, cuba, japan, u.s, america}.

Casos - HUM:ind - individuo humano

Caso HUM:ind - individuo humano

- Respuestas candidatas = Extraer NER tipo *PERSON* de las oraciones.
- Score: $0{,}5 \times \mathit{CoverageTarget} + 0{,}25 \times \mathit{CoverageSubject} + 0{,}35 \times \mathit{Sentence} + 0{,}25 \times \mathit{Proximity} + 0{,}1 \times \mathit{RepeatedTerm} + 0{,}5 \times \mathit{IsPronoun}$
  - CoverageTarget: cuántos tokens del target aparecen en la oración fuente de la respuesta candidata
  - CoverageSubject: cuántos tokens del subject aparecen en la oración fuente de la respuesta candidata (si se pudo encontrar un subject, cero si no - en nuestra adaptación es siempre cero)
  - Sentence: derivado de la posición de la oración en el ranking
  - Proximity: cuán cerca están la query utilizada como input del módulo de information retrieval (sin stop-words) y la respuesta candidata en el contexto de la oración de la que se extrajo la respuesta candidata.
  - RepeatedTerm: penalización (negativo). Coverage entre el target y la respuesta candidata
  - IsPronoun: Penaliza si la respuesta candidata contiene pronombres. En concreto, verifica si algún token pertenece a la lista {it, we, he, she, they, our, their}.

Casos - otros casos

HUM general y ENTY general: - otros temas humanos y otros objetos - Primer nombre propio de la oración mejor rankeada y se utiliza eso como respuesta.

LOC general: - preguntas que refieren a lugares - Se extraen todas las entidades nombradas de tipo 'LOCATION' de las oraciones rankeadas y se las evalúa según un score ponderado

Casos NUM - fechas, períodos, cuentas y números en general -

- La primer fecha encontrada según rank de pasajes (NUM:Date)
- El primer número encontrado según pasajes (NUM:Count)
- El primer numérico (número o fecha) para los otros casos NUM

Modificaciones

- Inferencia del tópico del grupo de preguntas
- Generación de queries
- Generación de respuestas

Inferencia del tópico del grupo de preguntas

- Grupos de 1 a 4 preguntas con tema
- Ejemplos: Colegio de Harry Potter, Pez Espada, Revolución de Terciopelo
- Condiciones del tema
  - Nombrado en primer pregunta/respuesta
  - Las siguientes preguntas pueden contener correferencias a este tópico
- Incorporado al sistema como "Target"
- Probamos diferentes valores:
  1. Test (el del test set)
  2. NERs + Numeros + Fechas
  3. Sustantivos
  4. 2 y 3
- 2, 3 y 4 basados solo en la 1er pregunta del cluster.

Generación de queries

1. Baseline / Qanus:
   - Tokens no repetidos ni stopwords de target (tópico)
   - Sustantivos, verbos y adjetivos de la pregunta completa
   - + números + fechas
2. Baseline "mejorado": *(TITLE: target)*^{n} *OR* ***query baseline***, $n = 5$
3. Lasso: como describimos en Estado de Arte
   1. Palabras no stop words entre comillas
   2. NERs
   3. Construcciones nominales con sus adjetivos
   4. Demás construcciones nominales
   5. Sustantivos con sus adjetivos
   6. Demás sustantivos
   7. Verbos
   8. ~~El focus de la pregunta~~

Generación de respuestas

- Scorers nuevos para rankear los pasajes
- No tocamos las heurísticas
- Scorers:
  - Length : Prioriza oraciones cortas pero no demasiado cortas.
  - QVerb: Presencia de verbos de la pregunta en la oración.
  - QNoun: Presencia de sustantivos.
  - QNER: Presencia de NERs.
- Tres métodos de rankeo de pasajes:
  $pscore_{bl} = 0{,}9 \times Coverage + 0{,}05 \times Freq + 0{,}05 \times Prox$;
  $pscore_2 = 0{,}4 \times pscore_{bl} + 0{,}2 \times QNER + 0{,}15 \times QVerb + 0{,}25 \times QNoun$
  $pscore_3 =$
  $0{,}4 \times pscore_{bl} + 0{,}15 \times Length + 0{,}1 \times QNER + 0{,}1 \times QVerb + 0{,}1 \times QNoun$

Experimentos

Corridas

- Idioma: español, portugués (2 opciones)
- Documentos retornados por Lucene: 50, 100, 200 (3 opciones)
- Pasajes extraídos de los documentos: 20, 40, 70 (3 opciones)
- Generación de Queries: baseline (1), improved baseline (2), lasso (3) (3 opciones)
- Inferencia de Temas: Test (1), NERs (2), sustantivos (3), híbrido (4) (4 opciones)
- Ranking de Pasajes: baseline (1), 2, 3 (3 opciones)

Evaluación de resultados

- Cantidad de respuestas dadas por el sistema: 1, 5, 10, 25 (4 opciones)
- Forma de evaluación automática: exacto, cubrimiento del 100 %, 75 % y 50 %, 15 % y 1 % (6 opciones)

Corrida 1: instanciación de cantidades

- Idioma: Español
- **Documentos retornados por Lucene: 50, 100, 200**
- **Pasajes extraídos de los documentos: 20, 40, 70**
- Generación de Queries: Baseline (1)
- Inferencia de Temas: NERs propio (2)
- Ranking de Pasajes: Baseline (1)

Corrida 1: resultados

Medida	Exacto	Covr 1	Covr .75	Covr .5	Covr .15	Covr .01
MRR_1	7.69	7.69	8.46	10.00	10.00	10.00
MRR_5	8.87	9.18	9.95	**12.51**	**13.47**	**13.54**
MRR_{10}	**9.37**	**9.44**	**10.21**	**12.77**	**14.25**	**14.31**
MRR_{25}	**9.56**	**9.63**	**10.40**	**13.07**	**14.58**	**14.65**

Medida	Exacto	Covr 1	Covr .75	Covr .5	Covr .15	Covr .01
MRR_1	**7.75**	**7.75**	**8.53**	**10.08**	**10.08**	**10.08**
MRR_5	**8.94**	**9.25**	**10.03**	12.42	13.39	13.45
MRR_{10}	9.31	9.38	10.16	12.55	14.03	14.10
MRR_{25}	9.54	9.61	10.39	12.89	14.41	14.48

Cuadro: Corrida 1: con 50 documentos de Lucene y 20, 40 pasajes

Observaciones y conclusiones

- 9 ejecuciones en total
- Más documentos de Lucene, peor en todas las permutaciones
- Muchos pasajes (70) también es malo en general
- 20 y 40 se comportan raro. $\rightarrow$ 40 es mejor para evaluaciones más exactas
- Fijamos 50 y 40 para el resto de las corridas

Corrida 2.1: Generación de Queries

- Idioma: Español
- Documentos retornados por Lucene: **50**
- Pasajes extraídos de los documentos: **40**
- **Generación de Queries: Baseline, Baseline mejorado, Lasso**
- Inferencia de Temas: NERs propio (2)
- Ranking de Pasajes: Baseline (1)

Corrida 2.1: Resultados

	Baseline		Baseline mejorado		Lasso	
Medida	Exacto	Covr 1	Exacto	Covr 1	Exacto	Covr 1
MRR_1	**7.75**	**7.75**	**3.91**	**3.91**	**7.81**	**7.81**
MRR_5	**8.94**	**9.25**	**5.36**	**5.91**	**9.27**	**9.58**
MRR_{10}	9.31	9.38	5.96	6.43	9.64	9.71
MRR_{25}	9.54	9.61	5.96	6.43	9.94	10.01

Cuadro: Corrida 2.1: Generación de Queries

Observaciones y Conclusiones

- Método lasso ligeramente superior a baseline
- Baseline "mejorado" rezagado siempre

Corrida 2.2: Inferencia de Temas

- Idioma: Español
- Documentos retornados por Lucene: 50
- Pasajes extraídos de los documentos: 40
- Generación de Queries: **Baseline**
- **Inferencia de Temas: Test (1), NERs (2), sustantivos (3), híbrido (4)**
- Ranking de Pasajes: Baseline (1)

Corrida 2.2: Resultados

	1) Test		2) NERs	
Medida	Exacto	Covr 1	Exacto	Covr 1
MRR_1	6.15	6.15	**7.75**	**7.75**
MRR_5	7.87	8.41	**8.94**	**9.25**
MRR_{10}	8.08	8.49	9.31	9.38
MRR_{25}	8.29	8.72	9.54	9.61

	3) Sustantivos		4) 2+3	
Medida	Exacto	Covr 1	Exacto	Covr 1
MRR_1	3.85	3.85	5.47	5.47
MRR_5	4.19	4.19	7.12	7.43
MRR_{10}	4.19	4.30	7.36	7.43
MRR_{25}	4.47	4.58	7.64	7.71

Cuadro: Corrida 2.2: Métodos 1, 2, 3 y 4 respectivamente

Conclusiones:

- El método utilizado (NERs + números de la primer pregunta como target) es el más efectivo en todos los casos.
- Esto es raro en relación al método 1

Corrida 2.3: Ranking de Pasajes

- Idioma: Español
- Documentos retornados por Lucene: 50
- Pasajes extraídos de los documentos: 40
- Generación de Queries: Baseline
- Inferencia de Temas: **NERs (2)**
- **Ranking de Pasajes: Baseline (1),** $pscore_2$**,** $pscore_3$

Corrida 2.3: Resultados

	$PassageScore_{bl}$		$PassageScore_2$		$PassageScore_3$	
Medida	Exacto	Covr 1	Exacto	Covr 1	Exacto	Covr 1
MRR_1	7.75	7.75	3.10	3.10	5.38	5.38
MRR_5	8.94	9.25	5.06	5.22	6.69	6.85
MRR_{10}	9.31	9.38	5.35	5.50	7.15	7.31
MRR_{25}	9.54	9.61	5.39	5.55	7.19	7.35

Cuadro: Corrida 2.3: Fórmulas 1, 2 y 3 respectivamente

Observaciones y conclusiones

- Todo mal
- diff $pscore_2$ y $pscore_3$ $\rightarrow$ *LengthScore*
  - Único agregado sustantivo a $pscore_3$
  - Aplicarlo al baseline
- Nuevas corridas con LenghtScore: 90/10 80/20 y 70/30
- La fórmula del score es la siguiente:

$$LengthScore = \begin{cases} 1{,}0 & 4 < \#tokens < 100 \\ 0{,}5 & 100 \leq \#tokens < 200 \\ 0{,}0 & \text{En cualquier otro caso} \end{cases}$$

Corrida 2.3: Resultados 2

	0.9 + 0.1		0.8 + 0.2		0.7 + 0.3	
Medida	Exacto	Covr 1	Exacto	Covr 1	Exacto	Covr 1
MRR_1	7.81	7.81	**10.08**	**10.08**	10.00	10.00
MRR_5	9.47	9.78	11.68	11.99	11.94	12.50
MRR_{10}	9.71	9.78	11.92	11.99	12.35	12.67
MRR_{25}	10.01	10.08	12.27	12.34	12.58	12.91

Cuadro: Corrida 2.3: Combinación ponderada de métodos 1 y 3

Observaciones y conclusiones

- El score funciona como filtro de pasajes mal formados o bien formados pero largos. En ambos casos, NLP pierde eficacia.
- Mejor resultado cuando más ponderacion salvo MRR_1 que se redujo en la tercer corrida.
  - Dado que esta métrica es la preferida, decidimos quedarnos con esta fórmula (8/2 y no 7/3)
- Mejoras fáciles acá.

Corrida 3: Combinación de Óptimos

Datos de la corrida 3.1, para español:

- Idioma: Español
- Documentos retornados por Lucene: 50
- Pasajes extraídos de los documentos: 40
- Generación de Queries: Baseline
- Inferencia de Temas: NERs (2)
- Ranking de Pasajes: $PassageScore_{bl} \times 0{,}8 + LengthScore \times 0{,}2$

Datos de la corrida 3.2, para portugués:

- Idioma: Portugués
- Documentos retornados por Lucene: 50
- Pasajes extraídos de los documentos: 40
- Generación de Queries: Baseline
- Inferencia de Temas: NERs (2)
- Ranking de Pasajes: $PassageScore_{bl} \times 0{,}8 + LengthScore \times 0{,}2$

Corrida 3: Resultados

Medida	Exacto	Covr 1	Covr .75	Covr .5	Covr .15	Covr .01
MRR_1	**10.65**	10.65	10.77	12.31	13.08	13.08
MRR_5	**12.00**	12.31	13.46	16.18	16.95	17.01
MRR_{10}	12.24	12.31	13.46	16.26	17.44	17.50
MRR_{25}	12.55	12.62	13.78	16.72	18.00	18.06

Cuadro: Corrida 3.1: Combinación de óptimos para español

Medida	Exacto	Covr 1	Covr .75	Covr .5	Covr .15	Covr .01
MRR_1	**1.92**	2.88	2.88	3.85	3.85	3.85
MRR_5	**3.16**	5.03	5.03	6.91	7.15	7.15
MRR_{10}	3.45	5.33	5.33	7.33	7.71	7.86
MRR_{25}	4.08	6.00	6.00	8.21	8.84	8.94

Cuadro: Corrida 3.2: Combinación de óptimos sobre portugués

Resultados de competidores de Clef '07

Run	% Overall Accuracy [200]
Priberam	44,5
Inaoe	34,5
Miracle	15
UPV	11,5
TALP	7

Figura: Competidores español

Run Name	Overall Accuracy (%)
diue071ptpt	40,9%
esfi071ptpt	7,4%
esfi072ptpt	4,0%
feup071ptpt	22,8%
ines071ptpt	11,4%
ines072ptpt	14,1%
prib071ptpt	61,7%

Figura: Competidores portugués

Corrida 3: observaciones y conclusiones

- Métodos que mejores resultados dieron para español
- Quizás, mal extrapolados a portugués
- Español peor que en 2.3 para muchas métricas
- Portugués: baja performance
  - Apenas un 3.16 % considerando la métrica MRR_5.
  - Las herramientas para portugués no son tan maduras como las herramientas para español;
  - Todo el desarrollo y el análisis estuvo basado en español
  - Solo POC de soporte para otros idiomas
- Español: 10.65 % exacto con una sola respuesta!
- No es el peor

- $10{,}65\,\% = \frac{13}{122}$
- $1{,}92\,\% = \frac{2}{104}$

Corrida 3: Respuestas ES

Q	A
¿De dónde es típica la ensaimada?	Mallorca
¿De cuántos bits era este procesador?	8 bits
¿En qué fecha El Corte Inglés compró Galerías Preciados?	24 de noviembre de 1995
¿Qué empresa preside Steve Jobs?	Apple Computer
¿Dé qué organización es comisionado David Stern?	NBA
¿Qué animales tiran del carro de Cibeles?	leones
¿Cuántos Premios Goya ganó "Torrente: El brazo tonto de la ley"?	dos
¿Quién fue la primera mujer que viajó al espacio?	Valentina Vladimírovna Tereshkova
¿A cuántos kilómetros de Huesca está el Castillo de Loarre?	35 kilómetros
¿Cuál es el nombre del presidente de Burundi que murió en 1994?	Cyprien Ntaryamira
¿Qué día se promulgó la constitución española de 1812?	19 de marzo
¿Qué dibujó Leonardo Da Vinci en 1492?	Hombre de Vitruvio
¿Qué programa de televisión presentó Takeshi Kitano de 1986 a 1989?	Humor Amarillo

Cuadro: Respuestas español

Corrida 3: Respuestas PT

Q	A
De que estado foi governador Adhemar de Barros?	São Paulo
Quando foi fundado o Nacional da Madeira?	8 de Dezembro de 1910

Cuadro: Respuestas portugués

Limitaciones y trabajo futuro

- Módulo multilingüe de Question Classification.
- Inferencia del tópico a partir del primer par de pregunta-respuesta.
- Generación del *focus*. Tema aparte.
- Preguntas de tipo definition y list, booleanas, cláusulas temporales.
- Mejorar las heurísticas

Conclusiones: Dominio abierto

- Sistema de question answering de dominio abierto con soporte para diferentes idiomas
- Qanus, Qa-sys monolingüe + Freeling + Lasso + Scorers
- CLEF'07 español y portugués + solo wikipedia + solo factoid + diferentes configuraciones
- CLEF'07: la mejor precisión general bajó del 49 % al 41.75 % para tareas multi-lingües, mientras que, más significativamente, bajó de 68 % a 54 % en tareas monolingües
- nos restringimos a una subsección sencilla (eliminando preguntas de tipo DEFINITION y LIST), por lo que no es del todo válido comparar esta performance con la nuestra.
- Qasys, el sistema baseline que adaptamos para soportar diferentes idiomas, sabemos que en la TREC 07 (en inglés), competencia en la que Qasys participó, el mejor sistema, LumbaPA07, obtuvo una exactitud de 70,06 %, mientas que el décimo (Quanta) obtuvo 20,6 % y Qasys logró un 11,9 %.
- Nuestro sistema adaptado al español, con todas las mejoras en su óptimo, logra un nada despreciable 10,08 para MRR_1 y un 12,00 para MRR_5.

Conclusiones 2

Sobre modelo de dominio abierto:

- Limitación de las mejoras a zonas acotadas y sin enfoques sistémicos.
- Una mejora argumentada y con cierta presencia en la literatura del área fue el mecanismo de generación de queries que llamamos Lasso (por el sistema en el que fue implementado por primera vez) y nosotros lo incorporamos aquí,
  - mejoras mínimas en la performance, de 7.75 a 7.81 en MRR_1 exacto (+0.26), de 8.94 a 9.27 en MRR_5 exacto (+0.31).
- Falta de "doctrina."[en] la generación de respuestas. Oportunidad! Linea muerta?
- Módulo de *focus* open source en inglés. Problema acotado, bien definido. Herramienta útil

¿Preguntas?