UNIVERSIDAD DE SANTIAGO DE CHILE

FACULTAD DE INGENIERÍA

DEPARTAMENTO DE INGENIERÍA INFORMÁTICA

Laboratorio N º2: Aplicación del Paradigma Lógico en un sistema simulador de chatbots

Nombre: Agustín Saavedra Olmos.

Asignatura: Paradigmas de Programación.

Profesor: Gonzalo Martínez Ramírez.

13 de Noviembre de 2023

TABLA DE CONTENIDO

CAPÍTULO 1: INTRODUCCIÓN

En el presente informe del laboratorio Nº2 se enfrenta una problemática sobre un sistema de simulación de chatbots, a través del uso del paradigma lógico en el lenguaje de programación Prolog. El objetivo general de esta experiencia es resolver el problema planteado usando la abstracción mediante los conceptos fundamentales del paradigma lógico. La estructura del informe contiene la introducción, la descripción del problema y del paradigma, diseño de la solución, aspectos de implementación, instrucciones de uso, resultados y autoevaluación, conclusiones, referencias bibliográficas y finalmente anexos.

CAPÍTULO 2: DESCRIPCIÓN DEL PROBLEMA

El problema planteado en este laboratorio consiste en crear, desplegar y administrar un sistema simulador de chatbots interactivo con el usuario, donde al sistema se le puede agregar chatbots, usuarios y chatHistory de los usuarios, a los chatbots se les puede agregar flujos y a los flujos se les puede añadir opciones mediante predicados constructores como option, flow y chatbot, también predicados modificadores como flowAddOption, chatbotAddFlow, entre otros predicados. Finalmente, se procede a simular y mostrar el sistema de chatbots con la finalidad de interactuar con el usuario con predicados como systemTalkRec, systemSynthesis y systemSimulate.

CAPÍTULO 3: DESCRIPCIÓN DEL PARADIGMA

El paradigma lógico está basado en el concepto matemático de la lógica de primer orden [1] y en las cláusulas de Horn [3] para poder inferir conclusiones a partir de datos. Esto quiere decir, que se establece una base de conocimientos a través de hechos y reglas, para después hacer una consulta a esa base de conocimientos [1]. Corresponde a un paradigma de programación declarativo, debido a que su enfoque es especificar lo que debe hacer un programa (¿QUÉ?).

Los conceptos que se aplican en el proyecto son los hechos, que corresponden a una cláusula que describe una relación entre términos que siempre son verdaderos en la base de conocimientos, los predicados que son relaciones

entre diferentes términos y se usan para declarar algo, la unificación que corresponde al mecanismo mediante el cual las variables lógicas toman valor [2], las consultas son las preguntas que se plantean a la base de conocimientos, para posteriormente usar la unificación para encontrar una "incógnita", si no se encuentra nada es falso [4] y por último las relaciones recursivas corresponde a la definición de los predicados a sí mismos, es decir que en el antecedente se consulta algún término que coincide con el predicado que está definiendo [4].

CAPÍTULO 4: ANÁLISIS DEL PROBLEMA

Para comenzar a abordar el problema, se identifican los TDA's básicos mínimos con los cuales se pueden hacer los requerimientos funcionales del proyecto. Son 6 TDA's necesarios para hacer el sistema de simulación de chatbots, los cuales corresponden a opción, flujo, chatbot, sistema, usuario y chatHistory.

Las opciones se encuentran compuestas por un código (entero), mensaje (string), código de enlace a un chatbot (entero), código de enlace a un flujo (entero) y palabras (lista de 0 o más palabras). Los flujos se componen de un id (entero), mensaje (string) y una lista de opciones (lista). A su vez, los chatbots contienen un id (entero), nombre (string), mensaje de bienvenida (string), id de flujo inicial (entero) y una lista de flujos (lista). El sistema de chatbots está compuesto por un nombre (string), id de chatbot inicial (entero), lista de chatbots (lista), lista de usuarios (lista), usuario ingresado en el sistema (lista) y el chatHistory del usuario ingresado en el sistema (lista).

En base a los datos anteriores, la problemática anterior se puede implementar en el paradigma lógico con una estructura de árboles debido a que hay una jerarquía entre los TDA's (Anexo Nº1), por ej: sistemas-chatbots, flujos-opciones, chatbots-flujos y la otra alternativa son las listas, las cuales son las estructura de datos más básicas en el lenguaje Prolog. En este caso, se escoge trabajar con listas ya que es más fácil de trabajar y manipular en esta estructura de datos para poder solucionar el problema planteado.

Se confirma que para trabajar en el Sistema se necesitan los TDA's opción, flujo, chatbot, sistema, usuario y ChatHistory para implementar funciones como systemAddUser (en la cual se agrega un Usuario creando una lista nueva,

considerando también los otras entradas del Sistema), systemLogout (donde se elimina el usuario creando una lista con los elementos del sistema, expresando lista vacía en el usuario registrado), etc.

CAPÍTULO 5: DISEÑO DE LA SOLUCIÓN

Por el análisis anterior se sabe que la estructura más importante de la problemática planteada es el TDA Sistema, debido a que contiene a todos los demás TDA's. Por otra parte, hay que considerar que la opción es la estructura más pequeña en el sistema de chatbots. Entonces, se debe crear la opción con un hecho que contiene todos los datos de la opción como variables y al final una lista con todo esos datos, además se hace el selector del código del TDA Opción, a través de una reglas usando el proceso de unificación. También se implementa el modificador que es agregar una opción a una lista de opciones, a través de un hecho que involucra listas.

Después se puede implementar el TDA Flujo creando su constructor y sus selectores de la misma forma que el anterior. El constructor y el modificador del TDA Flujo verifica si el id de las opciones está repetido o no, a través de la regla de remover opciones duplicadas (usando la función \+ member) que usa relaciones recursivas para ver los códigos repetidos y crear una nueva lista con las opciones que contienen distintos códigos.

En el TDA Chatbot se implementa el constructor verificando si los flujos contienen identificadores repetidos, sus selectores son en base a predicados de listas. Su modificador corresponde a chatbotAddFlow, este se implementa usando relaciones recursivas de forma natural a la repetición de identificadores de los flujos, en base a la función member.

Finalmente, el TDA Sistema sigue la mismas estructuras que las funciones anteriores, sin embargo sus modificadores systemAddChatbot y systemAddUser verifican los identificadores repetidos de una lista de chatbots y de una lista de usuarios usando en ambas relaciones recursivas de forma natural con reglas a través de los predicados remover chatbots duplicados y remover usuarios duplicados en un sistema de la misma forma que al remover opciones duplicadas en un flujo.

CAPÍTULO 6: ASPECTOS DE IMPLEMENTACIÓN

La estructura del proyecto consiste en 7 archivos correspondientes a las estructuras mínimas de la problemática planteada, las cuales son "option", "flow", "chatbot", "system", "user", "chatHistory" y por último el script de pruebas. Este último archivo, ejecuta los requerimientos de tipo funcional y muestra el funcionamiento del programa.

No se usa ninguna biblioteca externa para respetar lo más posible el paradigma lógico y así poder tener un mayor aprendizaje sobre los predicados, hechos, reglas y cláusulas del lenguaje en cuestión. El proyecto se desarrolla en el lenguaje de programación Prolog, a través de SWI-Prolog versión 9.0.4 y se usa el editor de código fuente Visual Studio Code versión 1.84.1.

CAPÍTULO 7: INSTRUCCIONES DE USO

Los archivos de cada TDA y el de script de pruebas deben estar en la misma carpeta para asegurar un buen funcionamiento del código y evitar cualquier tipo de error al momento de ejecutar el script de pruebas del laboratorio .

Es recomendable usar el comando “set_prolog_flag(answer_write_options, [max_depth(0)])” al momento de consultar las variables en Prolog para obtener una mejor visualización del resultado al momento de la ejecución. Es necesario usar el intérprete de SWI-Prolog al momento de realizar las consultas para poder utilizar el programa.

Se pueden generar errores posibles, si se introducen datos incorrectos o si no se sigue con los dominios de los predicados al momento de ejecutar el script de pruebas.

CAPÍTULO 8: RESULTADOS Y AUTOEVALUACIÓN

En cuanto a los resultados del requerimiento del proyecto, se logra cumplir las expectativas esperadas en los requerimientos funcionales y no funcionales (esto se ve en Anexo Nº2 y Anexo Nº3). El script de pruebas funciona y opera

de buena manera de manera total, incluso en los casos que se llegó a retornar false (ver Anexo Nº4). Los requerimientos funcionales que no se completaron son systemTalkRec, systemSynthesis y systemSimulate, debido a que ocurrieron problemas de abstracción que provocan la dificultad de entender lo que se pide en tales requerimientos.

CAPÍTULO 9: CONCLUSIONES

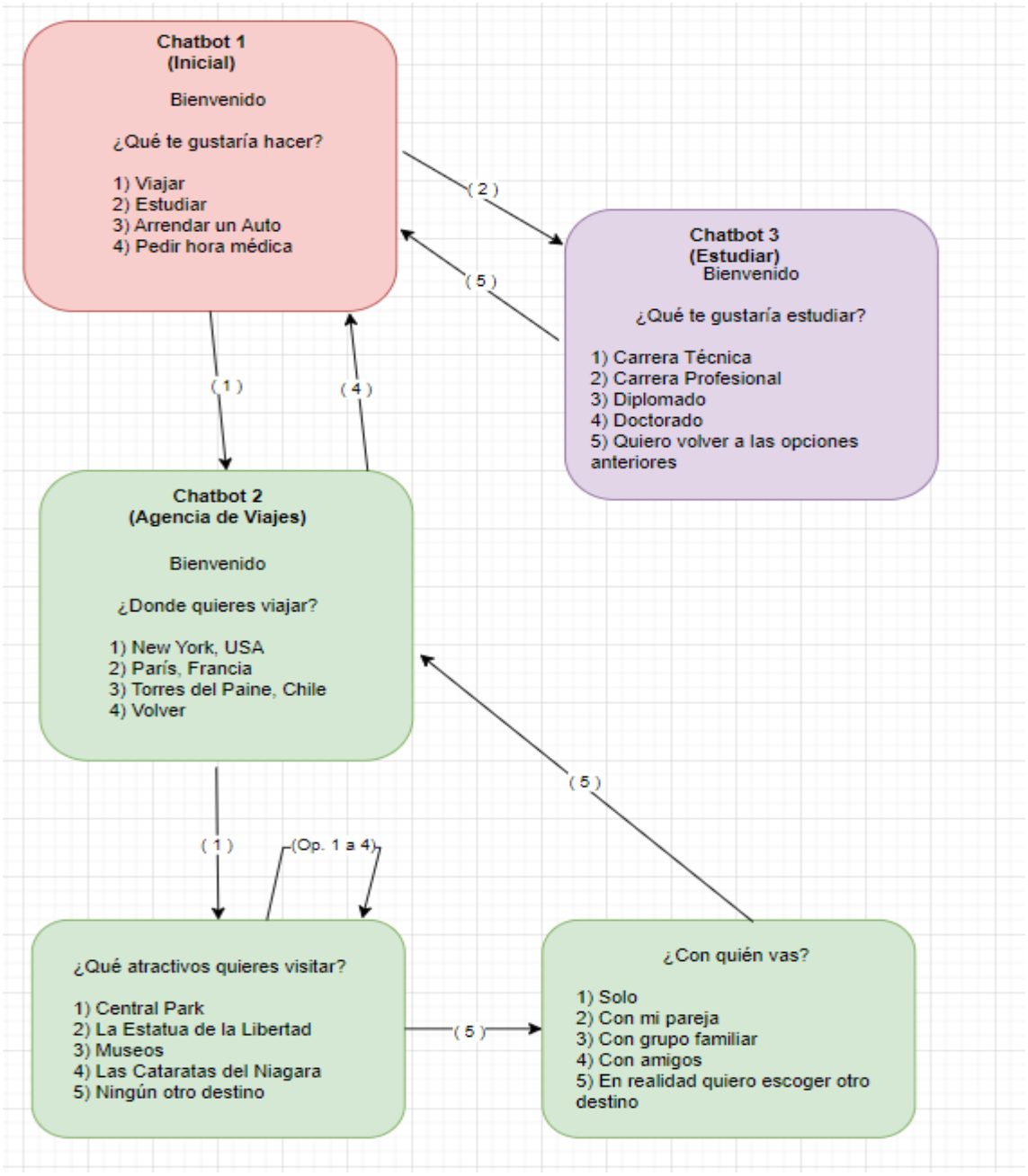
Respecto al grado de avance de los requerimientos, se cumple el objetivo propuesto de simular el sistema de chatbots a través de la abstracción usando los conceptos fundamentales del paradigma lógico. En comparación al paradigma funcional, el declarar hechos en vez de funciones y la forma de escribir el código hace mucho más fácil y óptimo el entendimiento y aprendizaje del problema.

La mayor dificultad del paradigma lógico es la manera de aplicar abstracción para poder conseguir predicados cómo systemTalkRec, systemSynthesis y systemSimulate como también entender las relaciones recursivas en el principio del aprendizaje en Prolog.

CAPÍTULO 10: REFERENCIAS

1. UAP. (Septiembre de 2015). Paradigma Lógico. *UAP.*https://blogramacion5.blogspot.com/2015/09/paradigma-logico.html.
2. *Tutorial de Programación en Prolog.* (s.f). https://www.dsi.fceia.unr.edu.ar/downloads/IIA/recursos/Tutorial%20de%20%20Prolog.pdf.
3. fedecarrion137. (19 de agosto de 2015). Paradigma Lógico. *Tu Paradigma.*https://tuparadigma.wordpress.com/2015/08/19/paradigma-logico/.
4. Gonzalez, R. (2023). 4.-P.Lógico. Paradigmas de Programación. UVirtual Usach. Recuperado de https://uvirtual.usach.cl/moodle/course/view.php?id=10036&section=16.

CAPÍTULO 11: ANEXOS



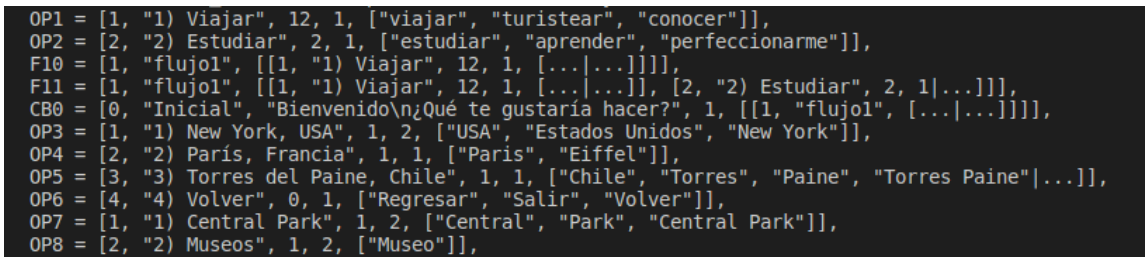
Anexo Nº1: Esquema de sistema de chatbots interactivo

#	Requerimiento No Funcional	Puntaje
1	Autoevaluación	1
2	Lenguaje	1
3	Versión	1
4	Standard	1
5	No variables	1
6	Documentación	1
7	Dom -> Rec	1
8	Organización	1
9	Historial	0.5
10	Script de Pruebas	1
11	Prerrequisitos	1

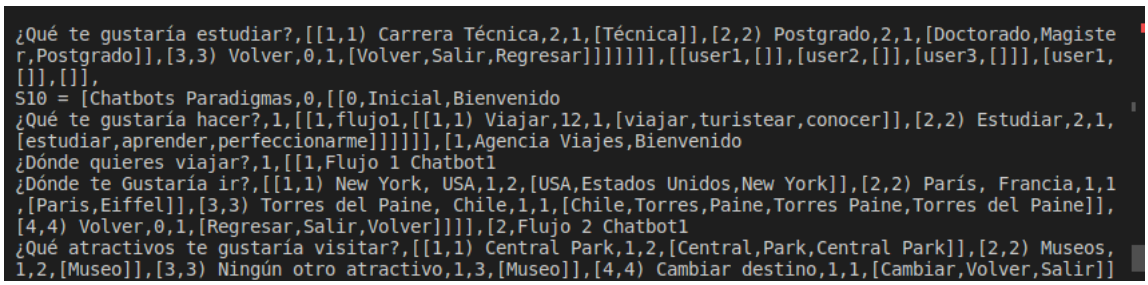
Anexo Nº2: Resultados de Requerimientos No Funcionales.

#	Requerimientos Funcionales	Puntaje
1	TDA's	1
2	option	1
3	flow	1
4	flowAddOption	1
5	chatbot	1
6	chatbotAddFlow	1
7	system	1
8	systemAddChatbot	1
9	systemAddUser	1
10	systemLogin	1
11	systemLogout	1
12	systemTalkRec	0
13	systemSynthesis	0
14	systemSimulate	0

Anexo Nº3: Resultados de Requerimientos funcionales.



```
OP1 = [1, "1) Viajar", 12, 1, ["viajar", "turistear", "conocer"]],
OP2 = [2, "2) Estudiar", 2, 1, ["estudiar", "aprender", "perfeccionarme"]],
F10 = [1, "flujo1", [[1, "1) Viajar", 12, 1, [...|...]]]],
F11 = [1, "flujo1", [[1, "1) Viajar", 12, 1, [...|...]], [2, "2) Estudiar", 2, 1|...]]],
CB0 = [0, "Inicial", "Bienvenido\n¿Qué te gustaría hacer?", 1, [[1, "flujo1", [...|...]]]],
OP3 = [1, "1) New York, USA", 1, 2, ["USA", "Estados Unidos", "New York"]],
OP4 = [2, "2) París, Francia", 1, 1, ["Paris", "Eiffel"]],
OP5 = [3, "3) Torres del Paine, Chile", 1, 1, ["Chile", "Torres", "Paine", "Torres Paine"|...]],
OP6 = [4, "4) Volver", 0, 1, ["Regresar", "Salir", "Volver"]],
OP7 = [1, "1) Central Park", 1, 2, ["Central", "Park", "Central Park"]],
OP8 = [2, "2) Museos", 1, 2, ["Museo"]],
```



```
¿Qué te gustaría estudiar?,[[1,1) Carrera Técnica,2,1,[Técnica]],[2,2) Postgrado,2,1,[Doctorado,Magiste
r,Postgrado]],[3,3) Volver,0,1,[Volver,Salir,Regresar]]]]]],[[user1,[]],[user2,[]],[user3,[]]],[user1,
[]],[]],
S10 = [Chatbots Paradigmas,0,[[0,Inicial,Bienvenido
¿Qué te gustaría hacer?,1,[[1,flujo1,[[1,1) Viajar,12,1,[viajar,turistear,conocer]],[2,2) Estudiar,2,1,
[estudiar,aprender,perfeccionarme]]]]],[1,Agencia Viajes,Bienvenido
¿Dónde quieres viajar?,1,[[1,Flujo 1 Chatbot1
¿Dónde te Gustaría ir?,[[1,1) New York, USA,1,2,[USA,Estados Unidos,New York]],[2,2) París, Francia,1,1
,[Paris,Eiffel]],[3,3) Torres del Paine, Chile,1,1,[Chile,Torres,Paine,Torres Paine,Torres del Paine]],
[4,4) Volver,0,1,[Regresar,Salir,Volver]]]],[2,Flujo 2 Chatbot1
¿Qué atractivos te gustaría visitar?,[[1,1) Central Park,1,2,[Central,Park,Central Park]],[2,2) Museos,
1,2,[Museo]],[3,3) Ningún otro atractivo,1,3,[Museo]],[4,4) Cambiar destino,1,1,[Cambiar,Volver,Salir]]
```

Anexo Nº4: Algunas pruebas de script en ejecución de programa.