**Guardian: empleo de sistemas de producción en una aplicación de seguridad.**

**Apellidos, Nombres (dejar esta parte en blanco en la versión de evaluación, pero sí escribirlos en la versión final)**

***Universidad Tecnológica Nacional, Facultad Regional Santa Fe***

**Abstract**

*En la cátedra IA se solicita a los alumnos que desarrollen un agente inteligente capaz brindar seguridad a su usuario.*

*Se considera que el usuario se encuentra en una situación de riesgo y el agente puede escuchar a través de un micrófono y transformarlo a texto para interpretar la situación del usuario. Si el agente piensa que el usuario está en peligro, emprende acciones para rescatarlo. En este trabajo se desarrolló el agente usando sistemas de producciones para la toma de decisión. Para que este sistema sea más eficiente se implementó el algoritmo de la red RETE. El sistema de producciones corre en Windows y el micrófono por el que el agente percibe es de un dispositivo con Android.*

**Palabras Clave**

Inteligencia artificial. Sistema de producciones. Red RETE. Seguridad.

**Introducción**

El presente trabajo se desarrolló como un práctico de la cátedra Inteligencia Artificial de la Facultad Regional Santa Fe. Se pidió el desarrollo de un agente que interprete las frases de un usuario o persona y descubra en la conversación patrones que estén asociadas a situaciones de peligro tales como "ayuda", "me están pegando", etc. Para dar solución al problema planteado por la catedra, se implementó un agente que transforma el sonido de la voz a texto y a través de un sistema de producciones el agente toma acciones que ayuden a la persona que está en problema, como por ejemplo llamar a la ambulancia, a la policía, etc.

1.  La reescritura cubre una amplia gama de métodos para reemplazar subtérminos de una fórmula por otros términos.

Este trabajo está relacionado con un trabajo previo realizado para la misma cátedra que permite encontrar un camino óptimo entre dos puntos de un mapa. Así la ambulancia o la policía utilizan estos algoritmos para encontrar el mejor camino a la ubicación del usuario.

En la implementación del sistema de producciones, para que el mismo sea eficiente, se implementó el algoritmo de red RETE de manera de mejorar los tiempos de respuesta.

Inicialmente, solo se solicitó que el sistema fuera capaz de recibir las palabras como texto plano. Posteriormente a la entrega del trabajo a la cátedra, se le adicionó la capacidad de captar audio del ambiente e interpretar palabras habladas utilizando el micrófono de un teléfono móvil y una aplicación de captura de voz.

Para entender los temas tratados en este paper es necesario ser familiar con los conceptos de ‘sistema de producciones’ y ‘red RETE’.

Los sistemas de producciones una forma de inteligencia artificial. Sus aplicaciones varían desde sistemas de reescritura de cadenas1 hasta sistemas expertos. Cuentan con una base de reglas de comportamiento compuestas, cada una, por una condición y una acción asociada. Tales reglas se conocen como ‘producciones’ y la base que las almacena es la ‘memoria de producciones’. Su estado está dado por una base de hechos llamada ‘memoria de trabajo’. La ‘máquina de inferencia’ del sistema de producciones verifica si los hechos cumplen las condiciones de las reglas y luego, si se cumplen las de varias, elige una y ejecuta su acción.

Una red RETE mejora la eficiencia de un sistema de producciones al recordar qué condiciones eran satisfechas en ciclos anteriores y actualizar esa información cuando cambian los hechos, sólo para las reglas afectadas. El algoritmo fue diseñado por Charles L. Forgy en 1983.

**Elementos del Trabajo y metodología**

Para resolver el problema presentado por la cátedra, se siguieron los siguientes pasos:

* Se identificaron los tipos de hechos que se almacenarían en la memoria de trabajo y las reglas que poblarían la memoria de producciones, usando la sintaxis de la lógica de primer orden.
* Se creó un listado de ‘palabras clave’ que al ser oídas advirtieran al sistema de un peligro potencial.
* Se programó un preprocesador de percepciones para filtrar de lo escuchado las palabras clave.
* Se programó el sistema de producciones.
* Se programó la interoperabilidad entre el sistema que es sujeto de este paper y la inteligencia artificial desarrollada en el trabajo previo de la cátedra.

A continuación, detallamos cómo se llevaron a cabo.

**Representación de reglas y hechos**

Se abstrajo el problema de la siguiente forma:

* Se eligieron seis tipos de incidentes ante los cuales el agente debiera reaccionar: delito callejero, delito en un hogar o tienda, violencia doméstica, emergencia médica, incendio y explosión. Cuando el agente reconociera alguno de estos incidentes, tomaría las acciones propicias, como por ejemplo llamar a la policía en caso de detectar un delito callejero.
* Hay palabras y frases que se suelen escuchar en contextos específicos, como cuando durante un robo. Se listaron, entonces, palabras y frases que pudieran relacionarse a los seis incidentes elegidos. Se les llamó a éstas ‘palabras clave’. A cada palabra clave se le asoció un valor numérico que representara cuán fuertemente estaban ligadas a un tipo de incidente.
* Existen subconjuntos de palabras clave que están muy fuertemente asociadas a un tipo particular de incidente. Se dice que estas palabras son “críticas” para identificar ese tipo de incidente, porque su percepción indica una gran probabilidad de que el incidente esté ocurriendo.
* Se reconoce con certeza que un incidente de un tipo particular está ocurriendo cuando se escucharon suficientes palabras clave asociadas a ese tipo de incidente y al menos una palabra crítica relacionada al mismo.
* Se considera que se escucharon suficientes palabras clave relacionadas a un incidente cuando la suma de los riesgos asociados a ellas supera un valor límite predefinido.

A partir de esta abstracción se identificaron reglas y clases de hechos. En la figura 2, se puede ver una de estas reglas.

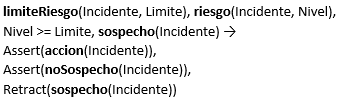


Figura 1: Esta regla es la que le indica al sistema que está ocurriendo un incidente y que debe actuar en respuesta.

**Palabras clave**

Las palabras clave y el tipo de incidente con el que se las podía relacionar se alimentaron al sistema de producciones como hechos, usando el formato de la figura 3. Éstas serían las palabras que el sistema podría reconocer.

No se le pasó, sin embargo, la familia entera de palabras de cada palabra clave, dado que eso hubiera resultado en una cantidad desmedida de hechos en la memoria de trabajo. Se seleccionó una sola forma de cada palabra. De los verbos, se eligió su forma infinitiva. De los sustantivos, su forma singular. Nos referiremos a ésta como la forma ‘normalizada’ de las palabras.



Figura 2: Este hecho se lee como “La palabra clave ‘Palabra’ está relacionada al tipo de incidente ‘TipoIncidente’ con una fortaleza de ‘Valor’”.

**Preprocesador de entrada**

El sistema de producciones reconocería, entonces, palabras clave en su forma normalizada, pero nada más. No sería capaz de reconocer sinónimos de las palabras clave, ni palabras no normalizadas.

De ahí es que se vio la necesidad de acoplar un módulo ‘preprocesador’ que tomara todo lo se escuchara, lo normalizara, transformara sinónimos de palabras clave en las palabras clave correspondientes y finalmente le pasara las palabras clave al sistema de producciones.

**Simulador basado en sistemas de producción.**

El sistema de producciones constituyó el grueso del trabajo realizado. Tanto éste como la red RETE que el mismo utiliza son de desarrollo propio.

Sus partes son:

* Máquina de inferencia (MI)

En implementaciones sin RETE, este módulo se encarga de verificar si las condiciones de las reglas son satisfechas por los hechos de la memoria de trabajo. En la implementación que se hizo para la cátedra, este trabajo es delegado a la memoria de producción.

La MI debe, por otro lado, seleccionar una regla a ejecutar dentro de todas las reglas cuyas condiciones se cumplen. Esto se implementó mediante la aplicación sucesiva de diversos criterios de resolución de conflicto, los cuales se definieron en el agente. Estos fueron ‘no duplicación’, ‘novedad’, ‘prioridad’, ‘especificidad’ y ‘aleatorio’, en ese orden.

Por último, debe ejecutar la regla. En la implementación que se hizo para el presente trabajo, el motor de inferencia devuelve una acción, la cual es ejecutada por el agente en el ambiente.

* Memoria de trabajo (MT)

Contiene los hechos. El estado del agente se implementó mediante una conexión a Prolog que al principio contiene hechos de inicialización, como los que listan las palabras clave y sus niveles de relación con incidentes, y luego va agregando y eliminando hechos a medida que se ejecutan las acciones. Cada vez que cambian los hechos que contiene la memoria de trabajo se les avisa a los nodos predicados de la red RETE de la memoria de producción y estos los propagan por la misma, activando y desactivando reglas.

* Memoria de producción (MP)

La memoria de producción se encarga del almacenamiento de reglas. Y es en este módulo donde la red RETE juega su papel principal.

**Red RETE tradicional**

Usualmente, una red RETE toma la forma de la Figura 3. En la ilustración se muestran la memoria de producción (MP) con las reglas, y la memoria de trabajo (MT) con los hechos. La red RETE está representada abajo, conformada por nodos ‘patrón’, nodos ‘join’ y nodos ‘terminales’. Los nodos patrón almacenan hechos que satisfacen predicados individuales de las condiciones. Los nodos join almacenan grupos de hechos que satisfacen grupos de predicados que involucran la misma variable. Por ejemplo, en la figura la primer y tercer regla tienen en sus condiciones A(x) y B(x). Ambas involucran a la misma variable ‘x’. El nodo join ‘A=B’ busca en los nodos patrón ‘A’ y ‘B’ coincidencias en el sujeto, y como se puede ver, tanto A como B aplican para el sujeto ‘2’. Entonces el nodo join ‘A=B’ almacena el par de hechos A(2) y B(2), indicando que ese par satisface al par A(x), B(x). Los nodos patrón y join se van conectando ‘a la derecha’ de los previos, satisfaciendo cada vez más requisitos de la condición hasta llegar a cumplir la condición completa, en cuyo punto se conectan con nodos terminales. Estos nodos almacenan grupos de hechos que en conjunto satisfacen la condición de una regla en su totalidad. Hay uno por cada regla. La máquina de inferencia solo necesita chequear los nodos terminales para saber qué reglas y, por lo tanto, qué acciones se pueden aplicar.

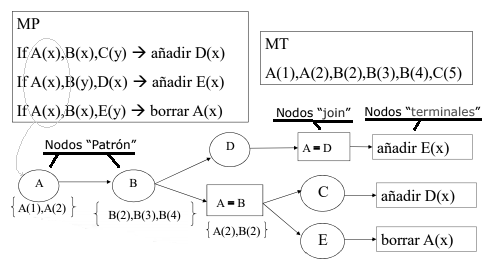


Figura 3: red RETE tradicional

La eficiencia de las redes RETE proviene del hecho de que, si los hechos no se modifican entre ciclo y ciclo, no se hacen nuevos cálculos. Y si se agrega o quita un hecho, el cambio se propagará desde el nodo patrón que le corresponde hasta los nodos terminales, habilitando o deshabilitando acciones con un mínimo de cálculos. El tiempo de procesamiento es, en teoría, independiente de la cantidad de reglas.

Por supuesto, las redes RETE tienen sus desventajas. La cantidad de memoria que ocupan es muy grande debido a la gran replicación de datos a lo largo de la red.

**La red RETE desarrollada**

La implementación de RETE que se hizo para el trabajo práctico cuenta con una mayor variedad de nodos que una red tradicional.

Para empezar, existen nodos ‘filtro’ que toman los hechos de otros nodos y los filtran bajo un criterio que depende de las condiciones. Por ejemplo, en la Figura 1 la condición solicitaba que el valor de ‘Nivel’ fuera mayor o igual al de ‘Limite’. Hay, entonces, un filtro específico que se encarga de propagar solo aquellos hechos que cumplan con ese requisito. Otro tipo de nodo filtro hace el papel de los nodos ‘join’ de la arquitectura tradicional.

Luego están los nodos ‘unión’, que no deben confundirse con los nodos ‘join’. Estos nodos reciben los hechos de otros nodos, y les aplica un ‘producto cartesiano’, generando todas las combinaciones posibles de esos hechos, que corresponderán a distintas formas de satisfacer la condición.

*Figura 4: Resultado de las pruebas de performance. Ambos ejes usan escala logarítmica*

Y existen versiones propias de los nodos ‘patrón’ y ‘terminales’, que para este trabajo se les llamó nodos ‘predicado’ y ‘regla’ respectivamente.

La red se inicializó inyectando los hechos presentes en la MT en los nodos ‘predicado’. Luego, se vinculó los mismos a la MT para que, al escuchar un cambio (adición o eliminación) de hechos asociados a los mismos, pudieran pedir y distribuir los mismos a través de la red.

**Integración con el trabajo práctico anterior**

El trabajo que es sujeto de este paper fue el segundo trabajo práctico realizado para la cátedra. El primero involucraba crear una inteligencia artificial para un patrullero que lo guiara por una ciudad para llegar a la escena donde ocurría un incidente, teniendo que sortear varios obstáculos.

El segundo trabajo práctico solicitó que la acción para alguna de las reglas fuera comunicarse con ese patrullero para enviarlo a la zona del incidente. Durante tal acción debía también generar el ambiente para el patrullero. La posición del incidente dentro de la ciudad se configuró para ser aleatoria en cada ejecución, mientras que la ciudad y los obstáculos serían siempre los mismos.

**Trabajo subsiguiente**

Posterior a la entrega del trabajo práctico a la cátedra, se le adicionaron funcionalidades nuevas al sistema.

En primer lugar, se programó una aplicación de móvil en Android que realmente es capaz de grabar sonido e interpretar lo que escucha usando una función del sistema operativo para reconocimiento del habla. La aplicación se conecta a un servidor que corre en pc y está comunicado al resto del sistema, el cual se ejecuta en la misma máquina.

En segundo lugar, se creó una interfaz de usuario con la cuál facilitar el agregado o la remoción de hechos a la memoria de trabajo.

**Resultados**

Las pruebas del sistema demostraron que éste es capaz de elegir acciones correctas cuando se lo alimenta con conversaciones cuyo contenido indican claramente que un incidente está ocurriendo. También pudo comunicarse con la inteligencia artificial del patrullero, el primer trabajo práctico para la cátedra de Inteligencia Artificial.

La performance del sistema de producciones es aceptable, gracias a la utilización de la red RETE.

**Discusión**

Un obstáculo para la realización de este trabajo es que todos los servicios de speech-to-text de escucha continua son pagos. Las capacidades del agente se encuentran actualmente limitadas por la aplicación de captura de audio que utiliza, la cual sólo graba por períodos cortos de tiempo. A pesar de esto, el agente está listo para funcionar con aplicaciones de escucha continua.

En cuanto a trabajo pendiente, el módulo responsable de normalizar las palabras de la entrada puede ser mejorado. Durante la realización del trabajo, la atención se focalizó en el sistema de producciones, por lo que se relegó la eficiencia del proceso de normalización.

**Conclusión**

El agente desarrollado cumple con las funciones solicitadas por la cátedra de IA. La red RETE permitió al agente ser eficiente y procesar un torrente continuo de palabras. La aplicación Android que se programó dotó al agente con funciones de captura de audio y speech-to-text.

En general, el trabajo fue una experiencia rica para sus participantes.

**Referencias**

1. Diapositiva ‘Sistemas de producciones’ de la cátedra Inteligencia Artificial. UTN FRSF.
2. https://www.infor.uva.es/~calonso/IAI/Tema9-Sistemas%20de%20Produccion/Rete.pdf
3. Klahr, D., Langley, P. and Neches, R. (1987). *Production System Models of Learning and Development*. Cambridge, Mass.

**Datos de contacto**

*Rico, Andrés. Institución: Universidad Tecnológica Nacional, Facultad Regional Santa Fe. Cód. postal: 3000. E-mail: andres.rico94@gmail.com*