UNIVERSIDAD
DE GRANADA

Ingeniería del Conocimiento
grado en ingeniería informática

PRÁCTICA FINAL

SBC para el riego automático de cultivos y plantas

Autor
Vladislav Nikolov Vasilev

DNI
X8743846M

Rama
Computación y Sistemas Inteligentes

Escuela Técnica Superior de Ingenierías Informática y de Telecomunicación

Curso 2018-2019

Índice

1. Resumen del funcionamiento del sistema

El sistema ha sido implementado utilizando **CLIPS** y **Python3**, más concretamente la versión 3.7.1 de Python. Para poder utilizar el sistema, se recomienda tener instalada una versión de Python igual o superior a la 3.7.1, además de tener instalados en el equipo los módulos *numpy* [1] y *scikit-fuzzy* [2], los cuáles son utilizados por el programa.

El sistema implementado en CLIPS es el sistema experto principal. Éste utiliza un pequeño programa escrito en Python que se encarga de realizar las deducciones correspondientes al riego de las plantas mediante un sistema basado en *lógica difusa*. Los dos programas se comunican enviándose la información necesaria para realizar las deducciones y las deducciones mismas mediante archivos.

El sistema principal se encuentra en *sistema.clp*, mientras que *sensores-init.clp* contiene información inicial relativa a los sensores (para no tener que insertar unos datos iniciales a mano) y *previsiones.clp* contiene una serie de previsiones del tiempo que se recomienda encarecidamente cargar en el sistema para que pueda funcionar de forma correcta, ya que el sistema supone que contiene información completa sobre la previsión del tiempo para las siguientes horasr. Estas previsiones pueden ser modificadas, tal y como ser indicará posteriormente. El archivo *fuzzy.py* contiene el sistema basado en lógica difusa que se encarga de realizar las deducciones para el riego de las plantas.

En general, el funcionamiento del sistema es bastante simple. El sistema experto va recibiendo valores de entrada de los sensores relativos a la **humedad**, **temperatura** y **luminosidad** de las plantas, además de valores de predicciones del tiempo. Con estos datos, el sistema decide si activar el **vaporizador** en aquellas plantas que considere oportunas con tal de reducir su temperatura o si activar el sistema de **riego** de las plantas, utilizando para ello los valores de los sensores. El sistema experto principal le envía los datos al sistema difuso para que se encargue de decidir cuál es el mejor curso de acción. Una vez decidido, envía los resultados de vuelta, y el sistema experto manipula esta información utilizando las predicciones de las próximas 8 horas con el objetivo de determinar cuál es el mejor curso de acción a tomar.

El sistema también permite pasar las horas una a una, simulando el paso del tiempo y actualizando el conocimiento de forma correspondiente, deduciendo por ejemplo si en la hora actual está lloviendo según la previsión que había para esa hora.

2. Descripción del proceso de creación del sistema

2.1. Procedimiento seguido para el desarrollo de la base de conocimiento

Para el desarrollo del sistema, se han ido satisfaciendo los requisitos especificados uno a uno. Estos requisitos son los siguientes:

1. Ser consciente de las necesidades de humedad de cada cultivo. Se ha de considerar, al menos, tres tiestos con rangos diferentes de humedad ideal.
2. Activar el riego automático de cada tiesto independientemente cuando dicho cultivo lo necesite.
3. Desactivar el riego de cada tiesto cuando se llegue a la humedad deseada.
4. Realizar un riego más inteligente en función del momento del día, considerando la posible evaporación.
5. Evitar, siempre que se pueda, el riego en situaciones de altas temperaturas y sol. No se podrá posponer en presencia de valores de humedad críticamente bajos.
6. Activar vaporizadores para refrescar el ambiente de cada tiesto y reducir altas temperaturas.
7. Posibilitar la combinación del uso de vaporizadores con el riego en momentos de altas temperaturas, cuando sea estrictamente necesario.
8. Utilizar la información meteorológica de la zona local para gestionar el riego de manera más eficiente, como posponer o regar con menor intensidad en caso de lluvia, según su intensidad y la certeza de la predicción.

Primero se ha comenzado especificando las plantas a utilizar, que en este caso han sido un **cactus**, un **tulipán** y una **palmera**, y se han especificado los valores mínimos y máximos de humedad, temperatura y luminosidad, además de los valores críticos de humedad y temperatura.

Sobre estos datos se ha añadido la capacidad de activar y desactivar el riego de forma automática para cada planta según su humedad. Posteriormente, se introdujo el sistema difuso escrito en Python3 que se encargaba de determinar si regar las plantas o no y con qué intensidad se deben regar en función de la información proporcionada por los sensores. Estas salidas se verán más adelante. Además de eso, se explicarán qué reglas ha seguido el sistema difuso para obtener las salidas.

Posteriormente, se añadió la capacidad al sistema de activar y desactivar los vaporizadores de las plantas, tal y cómo se hacía con el sistema de riego. Los vaporizadores tienen precedencia al riego, ya que si se usan antes de regar para reducir las temperaturas, se pueden conseguir unos riegos más eficaces, ya que se podrá regar más.

Finalmente, para poder trabajar con el tiempo atmosférico, se introdujo en el sistema la capacidad de representar el tiempo y las predicciones de tiempo según la hora, además del tiempo actual (si llueve o no y con qué intensidad lo hace). También se modificó el funcionamiento del sistema para poder manipular las salidas que le proporcionaba el sistema difuso para que, según el pronóstico de lluvias de las próximas 8 horas, pueda ser capaz de deducir si se debe regar menos de lo que indica la salida o si se debería proceder tal y como ha indicado el sistema difuso. En secciones posteriores veremos cómo se han representado estos hechos y cómo se han ido deduciendo las cosas en el sistema.

2.2. Procedimiento de validación y verificación del sistema

La validación y verificación del sistema ha sido un proceso que se ha ido haciendo a lo largo de todo el desarrollo, desde las etapas iniciales hasta las finales.

2.2.1. Verificación

En cuanto a la verificación, se ha comprobado que el sistema fuese el correcto en cada etapa. A medida que se iba añadiendo funcionalidad al sistema se iba comprobando que ésta funcionase correctamente. Por ejemplo, si se incluía una nueva regla, se probaba de forma interactiva añadiendo hechos a la base de conocimiento con el objetivo de ver si esa regla nueva era capaz de activarse o no. También se ha ido comprobando a medida que se iba construyendo el sistema que no hubiesen **errores sintácticos** ni **inconsistencias**, como por ejemplo reglas que nunca se llegasen a disparar o que produjesen resultados contradictorios entre ellas. También se ha comprobado que no se produjesen inconsistencias semánticas con el fin de mantener la coherencia dentro del sistema. Se ha ido comprobando, por ejemplo, si los valores deducidos se correspondían con el tipo que debían tener.

Otra cosa que se ha hecho para validar el sistema es simular el comportamiento de un usuario para ver que podía trabajar con normalidad sin que se produjesen errores o inconsistencias.

Con todo el proceso de verificación se cree que se han eliminado la mayoría o casi todos los errores, ya que se ha ido haciendo de forma progresiva. Sin embargo,

nunca podemos estar seguros del todo, con lo cuál es posible que aún exista algún que otro pequeño error que no hayamos podido detectar debido a que, por motivos de tiempo, es imposible simular toda la actividad que realizaría un usuario, ya que existen infinitos casos que se pueden dar.

2.2.2. Validación

En cuanto a la validación, se ha ido comprobando en cada fase que se cumplían los requisitos especificados. Además, se ha comprobado que los resultados obtenidos fuesen correctos según las entradas que se le iba proporcionando al sistema. Aparte de esto, se ha comprobado que la comunicación entre el sistema principal y el sistema difuso fuese correcta, que fuese un sistema relativamente rápido y que fuese capaz de funcionar en tiempo real y que, además de todo esto, el sistema explicase suficientemente bien todo lo que sucede en el sistema, explicando al usuario todas las deducciones que va haciendo de tal forma que éste tenga en todo momento suficiente información para saber qué es lo que está ocurriendo.

Se cree que, en general, los resultados que ofrece el sistema son buenos. Sin embargo, al no haber extraído el conocimiento de un experto si no de nosotros mismos, ya que nos hemos considerado los expertos, puede que hayamos cometido algún error a la hora de, por ejemplo, especificar con qué intensidad se debe regar una planta en función de la información proporcionada por los sensores de entrada. Esto es inevitable, ya que no tenemos una fuente real de conocimiento experto. Sin embargo, el sistema está hecho de tal forma que se pueda modificar el funcionamiento fácilmente.

Por ejemplo, si no se está de acuerdo con algún resultado obtenido de, por ejemplo, el riego que se debe realizar, se puede revisar el sistema difuso y modificar las reglas del riego para que produzcan unos resultados que se ajusten más a los que deberían obtenerse realmente o a los deseados por el usuario.

3. Descripción del sistema

3.1. Variables de entrada del problema

Las entradas al sistema son los valores que miden los sensores de humedad, temperatura y luminosidad para cada planta y la previsión de lluvia para una determinada hora.

Las entradas de los sensores tienen la siguiente forma:

```
(valor Tipo Planta Valor-Medido)
```

donde:

- *Tipo* representa Humedad, Temperatura o Luminosidad.
- *Planta* hace referencia a una de las tres plantas (Cactus, Tulipan o Palmera).
- *Valor-Medido* es el valor medido por el sensor. Todos son valores enteros y toman los siguients valores:
  - En el caso de la **humedad**, los valores posibles están en el rango $[0, 1023]$, donde 0 indica humedad máxima y 1023 indica lo más seco posible.
  - En el caso de **temperatura**, los valores posibles son todos los enteros, tanto positivos como negativos. Sin embargo, el sistema difuso solo acepta valores en el rango $[0, 99]$, ya que no parece muy posible que las plantas que se cuiden sobrevivan a temperaturas superiores o inferiores a esas.
  - En el caso de **luminosidad**, los valores van desde 0 a cualquier valor, no hay límite. Sin embargo, el sistema difuso asume que la luminosidad más grande es 1000. De ser necesario, se puede ampliar.

Las entradas de las previsiones tienen la siguiente forma:

```
(prevision_lluvia Hora Intensidad)
```

donde:

- *Hora* es un valor entero en el rango $[0, 23]$ que representa a qué hora hace referencia la previsión.
- *Intensidad* es un valor real que indica cuál es la intensidad de la lluvia en mm/h.

Estos datos de entrada son procesados para facilitarle el funcionamiento al sistema posteriormente. Para cada **valor** de entrada, se generan estos dos hechos:

```
(valor_registrado Tiempo Tipo Planta Valor-Medido)
(utlimo_registro Tipo Planta Tiempo)
```

donde *Tiempo* es un valor que representa un tiempo desde que se inició el sistema. Esta es una forma de identificar a los valores registrados y hacerlos únicos. Así, desde el **ultimo_registro** se puede acceder fácilmente al *Valor-Medido* utilizando *Tiempo*.

Para las previsiones, se obtienen los siguientes datos de entrada procesados:

```
(LluviaPrevista Hora Tipo Intensidad)
```

donde *Tipo* va en función de la intensidad. Es decir:

- Si *Tipo* es **no** es que no va a llover, lo cuál significa que *Intensidad* está en el rango $[0, 0.2)$.
- Si *Tipo* es **debil** es que habrá lluvia de intensidad débil, lo cuál significa que *Intensidad* está en el rango $[0.2, 6.5)$.
- Si *Tipo* es **moderada** es que habrá lluvia de intensidad moderada, lo cuál significa que *Intensidad* está en el rango $[6.5, 15)$.
- Si *Tipo* es **fuerte** es que habrá lluvia de intensidad fuerte, lo cuál significa que *Intensidad* está en el rango $[15, 100)$.
- Si *Tipo* es **torrencial** es que habrá lluvia de intensidad torrencial, lo cuál significa que *Intensidad* está en el rango $[100, \infty)$.

3.2. Variables de salida del problema

Como tal, el sistema no produce ningún valor de salida. Sin embargo, tiene que interactuar con los sistemas de riego y vaporización. Para ello, el sistema tiene que activar/desactivar estos sistemas, con lo cuál usará una serie de hechos.

Para el caso del riego, se tiene el siguiente hecho:

```
(regar Planta Estado [HumedadObjetivo])
```

donde se tiene que:

- *Planta* es la planta asociada a dicho sistema de riego.

- *Estado* indica si el sistema de riego está encendido o apagado mediante los valores on/off.
- *HumedadObjetivo* es un valor que solo aparece si *Estado* está en on. Este valor indica cuál es la humedad que se quiere conseguir con el riego.

En cuanto a la vaporización, tenemos el siguiente hecho:

```
1 (vaporizador Planta Estado [TemperaturaObjetivo])
```

donde se tenemos:

- *Planta* es la planta asociada a dicho sistema de vaporización.
- *Estado* indica si el sistema de vaporización está encendido o apagado mediante los valores on/off.
- *TemperaturaObjetivo* es un valor que solo aparece si *Estado* está en on. Este valor indica cuál es la temperatura a conseguir con la vaporización.

3.3. Conocimiento global del sistema

El sistema tiene una serie de hechos y reglas que son globales para todo él. Es decir, pueden ser accedidas desde cualquier módulo

3.4. Especificación de los módulos desarrollados

3.4.1. Estructura de los módulos

3.4.2. Descripción de los módulos

3.5. Hechos y reglas de cada módulo

A continuación se pueden ver las reglas utilizadas por el sistema:

si $Humedad(Baja) \wedge Luminosidad(Baja) \wedge Temperatura(Baja) \rightarrow$ regar media
si $Humedad(Baja) \wedge Luminosidad(Baja) \wedge Temperatura(Media) \rightarrow$ regar alta
si $Humedad(Baja) \wedge Luminosidad(Baja) \wedge Temperatura(Alta) \rightarrow$ regar media
si $Humedad(Baja) \wedge Luminosidad(Media) \wedge Temperatura(Baja) \rightarrow$ regar media
si $Humedad(Baja) \wedge Luminosidad(Media) \wedge Temperatura(Media) \rightarrow$ regar alta
si $Humedad(Baja) \wedge Luminosidad(Media) \wedge Temperatura(Alta) \rightarrow$ regar poca
si $Humedad(Baja) \wedge Luminosidad(Alta) \wedge Temperatura(Baja) \rightarrow$ regar poca
si $Humedad(Baja) \wedge Luminosidad(Alta) \wedge Temperatura(Media) \rightarrow$ regar media
si $Humedad(Baja) \wedge Luminosidad(Alta) \wedge Temperatura(Alta) \rightarrow$ regar poca
si $Humedad(Media) \vee Humedad(Alta) \rightarrow$ no regar

Las salidas que devuelve son con qué intensidad regar en los casos en los que se debe hacer (**poca**, **media** o **alta intensidad**) o si no debe hacerse.

4. Manual de uso del sistema

Referencias

[1] NumPy
`https://www.numpy.org/`

[2] GitHub. *scikit-fuzzy*
`https://github.com/scikit-fuzzy/scikit-fuzzy`