XVII Simposio CEA de Control Inteligente
27-29 de junio de 2022, León

Aprendizaje por refuerzo profundo para la negociación en control predictivo distribuido cooperativo de sistemas multivariables

Aponte, O.E.[1,*], Vega, P.[1], Francisco, M.[1]

[1] *Departamento de Informática y Automática, Facultad de Ciencias, Universidad de Salamanca, Plaza de la Merced, s/n, 37008, Salamanca, España.*

To cite this article: Aponte, O.E., Vega, P., Francisco, M. 2022. Deep reinforcement learning for negotiation in cooperative distributed predictive control of complex systems. XVII Simposio CEA de Control Inteligente.

Resumen

En este trabajo se propone una solución novedosa del uso de redes neuronales con aprendizaje por refuerzo, como una opción válida en la negociación de agentes de controladores jerárquicos distribuidos. El método propuesto se implementa en la capa superior de una arquitectura de control jerárquico constituido en sus niveles más bajos por un control distribuido basado en modelos locales y procesos de negociación con lógica difusa. La ventaja de la propuesta es que no requiere el uso de modelos en la negociación y facilita la minimización de cualquier índice de comportamiento dinámico y la especificación de restricciones. Concretamente, se utiliza un agente de gradiente de políticas (PG) del aprendizaje por refuerzo para el entrenamiento del proceso de consenso entre los agentes. El algoritmo se aplica con éxito a un sistema de nivel compuesto por ocho tanques interconectados muy difícil de controlar por su naturaleza no lineal y la alta interacción existente entre sus subsistemas.

Palabras clave: Redes neuronales, Aprendizaje por refuerzo, control cistribuido predictivo basado en modelos, sistemas multiagente.

Deep reinforcement learning for negotiation in cooperative distributed predictive control of complex systems

Abstract

In this work, a novel solution of the use of neural networks with reinforcement learning is proposed, as a valid option in the negotiation of distributed hierarchical controller agents. The proposed method is implemented in the upper layer of a hierarchical control architecture constituted in its lower levels by a distributed control based on local models and fuzzy logic negotiation processes. The advantage of the proposal is that it does not require the use of models in the negotiation and facilitates the minimization of any dynamic behaviour index and the specification of constraints. Specifically, in this work, a reinforcement learning policy gradient (PG) algorithm is used to train the consensus process between agents. The resulting algorithm is successfully applied to a level system made up of eight interconnected tanks, which is very difficult to control due to its non-linear nature and the high level of interaction between its subsystems.

Keywords: Neural networks, Reinforcement learning, fuzy logic, distributed model predictive control, multiagent systems

1. Introducción

Cuando los requisitos computacionales y de comunicación involucrados en un problema de control centralizado predictivo por modelo, en inglés "Model predictive control"(MPC), son inviables desde un punto de vista práctico, una posible solución es utilizar MPC distribuidos, en inglés "Distributed model predictive control"(DMPC), para dividir el problema en varios más simples (Cembellín et al., 2019). Entre ellos los basados en sistemas multiagente y negociación son una alternativa útil que ha sido la empleada en este trabajo (Francisco et al., 2019; Masero et al., 2021). La tarea crítica para resolver el problema del consenso en sistemas multiagentes es diseñar protocolos de control distribuido basados en la información de cada agente y sus vecinos. Además, en el campo de control, es de vital importancia el uso de estrategias que permitan a los agentes cambiar

* Autor para correspondencia: idu17344@usal.es

sus decisiones en línea, de acuerdo con los cambios en el entorno o el comportamiento de otros agentes con el fin de lograr un acuerdo conjunto sobre un interés común.

Las técnicas adecuadas que se pueden aplicar en este contexto son las metodologías de redes neuronales basadas en el aprendizaje por refuerzo, en inglés Reinforcement Learning"(RL), que ofrecen muchas ventajas y que han motivado el desarrollo de muchos algoritmos para la toma de decisiones y el control, proporcionando el estudio de todas las propiedades de estabilidad, convergencia y viabilidad que implica un problema de control (Sierra-Garcia et al., 2022; Aponte et al., 2022). Precisamente, uno de los principales objetivos de este proyecto es proponer nuevas técnicas basadas en aprendizaje por refuerzo para resolver en los juegos multijugador en tiempo real los problemas que surgen en el control distribuido cooperativo.

El aprendizaje por refuerzo tiene como objetivo principal la capacitación de un agente para que complete una tarea en un entorno incierto, esta capacitación se lleva a cabo a través de un entrenamiento que consiste en la interacción agente-entorno. El agente aprende por medio de ensayo y error, enviando una acción al entorno, y recibiendo de este, un estado y recompensa, ambas, consecuencia de la acción (Sutton and Barto, 2018). El estado indica la influencia de la acción sobre el entorno, mientras que la recompensa califica a la acción.

El objetivo global de este trabajo es el desarrollo de una metodología (PG DMPC) de control multiagente que permita el uso de técnicas pertenecientes al campo del DMPC y el agente PG del aprendizaje por refuerzo para implementar los procesos de negociación entre agentes. Concretamente, se realiza en este trabajo una propuesta novedosa de aprendizaje por refuerzo en las capas superiores de la jerarquía de control y su aplicación a un sistema multivariable de control no lineal (Masero et al., 2021).

El resto del artículo está organizado de la siguiente manera. En la sección dos se describen los algoritmos de control, el método de negociación del que se parte y el método propuesto. La sección tres describe el agente de aprendizaje por refuerzo, la cuarta sección el caso de estudio. El quinto apartado contiene la implementación del método. En el sexto se presentan los resultados del entrenamiento y validación, en el séptimo apartado los resultados, y en octavo las conclusiones.

Notación: La secuencia de entrada de dimensión N_p en cada instante de tiempo t se define como $U(t) = [u(t), u(t+1), ..., u(t+N_p-1)]^\intercal$ y la secuencia óptima es $U^*(t) = [u^*(t), u^*(t+1), ..., u^*(t+N_p-1)]^\intercal$.

2. Formulación del problema

Estando el sistema compuesto por un conjunto $\mathcal{N} = \{1, 2, ..., \mathbf{N}\}$ de subsistemas acoplados a la entrada cuya evolución de estado es

$$x_i(t+1) = A_i x_i(t) + B_{ii} u_i(t) + w_i(t), \tag{1}$$

donde $t \in \mathbb{N}_0+$ denota el instante de tiempo; $x_i \in \mathbb{R}^{q_i}$ y $u_i \in \mathbb{R}^{r_i}$ son, respectivamente, los vectores de estados y entradas de cada subsistema $i \in \mathcal{N}$, restringido en los conjuntos convexos que contienen el origen en su interior $\mathcal{X}_i = \{x_i \in \mathbb{R}_{q_i} : A_{x,i} x_i \leqslant b_{x,i}\}$ y $\mathcal{U}_i = \{u_i \in \mathbb{R}_{r_i} : A_{u,i} u_i \leqslant b_{u,i}\}$, respectivamente; y $A_i \in \mathbb{R}^{q_i x q i}$ y $B_{ii} \in \mathbb{R}^{q_i x r i}$ son matrices de dimensiones propias. El vector de perturbaciones medibles $w_i \in \mathbb{R}^{q_i}$ representa el acoplamiento con otros subsistemas j perteneciente al conjunto de vecinos $\mathcal{N}_i = \{j \in \mathcal{N} \setminus \{i\} : B_{ij} \neq 0\}$, es decir,

$$w_i(t) = \sum_{j \in \mathcal{N}_i} B_{ij} u_j(t), \tag{2}$$

donde $u_j \in \mathbb{R}^{r_j}$ es el vector de entrada del subsistema $j \in \mathcal{N}_i$, y la matriz $B_{ij} \in \mathbb{R}^{q_i \times r_j}$ modela el acoplamiento de entrada entre i y j. Es más, w_i está acotado en un conjunto convexo $\mathcal{W} = \oplus_{j \in \mathcal{N}} B_{ij} \mathrm{U}_j$ debido a las limitaciones del sistema. El vecino afectado por la agente i se define como $\mathcal{M}_i = \{j \in \mathcal{N} \setminus \{i\} : B_{ji} \neq 0\}$. Desde el punto de vista global, la evolución general del sistema se puede definir como

$$x_{\mathcal{N}}(t+1) = A_{\mathcal{N}} x_{\mathcal{N}}(t) + B_{\mathcal{N}} u_{\mathcal{N}}(t), \tag{3}$$

donde $A_{\mathcal{N}} = [A_{ij}]_{ij \in \mathcal{N}}$ y $B_{\mathcal{N}} = [B_{ij}]_{ij \in \mathcal{N}}$ son, respectivamente, las matrices de estados y entradas de todo el sistema.

Siendo el objetivo, obtener el vector de entrada $U_{\mathcal{N}}$ ponderado a través de negociación por pares empleando un coeficiente ϕ generado por una red neuronal profunda bajo el formato de aprendizaje por refuerzo.

En cada instante de tiempo t, la función de costo del subsistema $i \in \mathcal{N}$ se calcula con base en las trayectorias de sus estados y entradas sobre una ventana futura de longitud N_p (el llamado horizonte de predicción):

$$\begin{aligned} &J_i(x_i(t), U_i(t), U_j(t)) \\ &= \textstyle\sum_{t=0}^{N_p-1} L_i(x_i(t+k), u_i(t+k)) + \\ &F_i(x_i(t+N_p)), \end{aligned} \tag{4}$$

donde $L_i(\cdot)$ es la función de costes de estado, y $F_i(\cdot)$ la función de coste terminal, definidas como

$$\begin{aligned} &L_i(x_i(t+k), u_i(t+k)) = \\ &(x_i(t+k) - x_{r_i}(t+k))^\intercal Q_i x_i(t+k) \\ &+ (u_i(t+k) - u_{r_i}(t+k))^\intercal R_i (u_i(t+k) - u_{r_i}(t+k)), \\ &F_i(x_i(t+k)) = (x_i(t+N_p) - x_{r_i}(t+k))^\intercal \\ &P_i(x_i(t+N_p) - x_{r_i}(t+k)), \end{aligned} \tag{5}$$

siendo Q_i una matriz definida semipositiva, y R_i, P_i siendo matrices definidas positivas. Siendo x_{r_i} y u_{r_i}, las referencias calculadas para obtener un error estacionario nulo según (Francisco et al., 2015).

3. Arquitectura de control

La comunicación entre los agentes se realiza mediante una red que se puede modelar como el grafo indirecto $\mathcal{G} = (\mathcal{N}, \mathcal{L})$ donde $\mathcal{N}$ es el conjunto de agentes y $\mathcal{L}$ es el conjunto de enlaces bidireccionales $\mathcal{L} \subseteq \mathcal{L}^{\mathcal{N}} = \{\{i, j\} : \{i, j\} \subseteq \mathcal{N},\ i \neq j\}$ (Maestre et al., 2011). Un enlace $l_{ij} \in \mathcal{L}$ conecta los agentes i y j dando un flujo de información bidireccional. Si dos agentes están conectados por un enlace de comunicación l_{ij} siguen un esquema cooperativo por pares para encontrar un consenso entre sus secuencias de control a través de un algoritmo multicapa de negociación de lógica difusa en la primera capa.

3.1. Capa de control de bajo nivel

La primera capa de control emplea un algoritmo DMPC para múltiples agentes (Masero et al., 2021). Las negociaciones con lógica difusa se realizan en pares teniendo en cuenta los acoplamientos con los subsistemas vecinos. Para ello, se emplea una secuencia desplazada de agente *i*, que es definida añadiendo $K_i x_i(t+N_\mathrm{p})$ a la secuencia elegida en el paso previo $U_i(t-1)$:

$$U_i^\mathrm{s}(t) = \begin{bmatrix} u_i(t+1|t-1) \\ u_i(t+2|t-1) \\ \vdots \\ u_i(t+N_\mathrm{p}-1|t-1) \\ K_i x_i(t+N_\mathrm{p}|t-1) \end{bmatrix} = \begin{bmatrix} u_i^\mathrm{s}(t) \\ u_i^\mathrm{s}(t+1) \\ \vdots \\ u_i^\mathrm{s}(t+N_\mathrm{p}-2) \\ u_i^\mathrm{s}(t+N_\mathrm{p}-1) \end{bmatrix}. \tag{6}$$

El algoritmo empleado, Algoritmo 1, es una extensión del esquema DMPC propuesto por (Maestre et al., 2011) para *n* subsistemas.

Algoritmo 1: Algoritmo DMPC.

Para cada paso de tiempo *t*:

1. En cada instante *t* cada agente *i* mide $y_i(t)$ y estima el estado inicial para la predicción del MPC.
2. El agente *i* calcula su trayectoria desplazada y la envía a sus vecinos.
3. Cada agente *i* minimiza su función de costo J_i teniendo en cuenta que su vecino *j* aplica su trayectoria desplazada. Los otros subsistemas vecinos siguen sus trayectorias de control actuales $U_j^\mathrm{s}(t)$. Específicamente, el agente *i* resuelve:

$$U_i^*(t) = \arg\min_{U_i(t)} J_i\big(x_i(t), U_i(t), U_j^\mathrm{s}(t), U_l^\mathrm{s}(t)\big), \tag{7}$$

Sujetos a las siguientes restricciones sobre entradas, estados y restricción de estado terminal Ω_i:

$$\begin{aligned} &x_i(t+k+1) = A_i x_i(t+k) + B_{ii}u_i(t+k) + \\ &B_{ij}u_j(t+k) + \sum_{l\in\mathcal{N}_i\setminus\{j\}} B_{il}u_l(t+k), \end{aligned} \tag{8}$$

$$\begin{aligned} &x_i(t) = \tilde{x}_i(t),\ i\in\mathcal{N}, \\ &x_i(t+k)\in\mathcal{X}_i, && k=0,\ \ldots, N_\mathrm{p}-1, \\ &x_i(t+N_\mathrm{p})\in\Omega_i, \\ &u_i(t+k)\in\mathcal{U}_i, && k=0,\ \ldots, N_\mathrm{p}-1, \\ &u_j(t+k) = u_j^\mathrm{s}(t+k), && k=0,\ \ldots, N_\mathrm{p}-1, \\ &u_l(t+k) = u_l^\mathrm{s}(t+k), && k=0,\ \ldots, N_\mathrm{p}-1, \end{aligned} \tag{9}$$

donde se establece Ω_i se impone como restricción de estado terminal del agente, detalles sobre el calculo de Ω_i se pueden ver en(Masero et al., 2021).

4. El agente *i* optimiza nuevamente su costo $J_i(\cdot)$ manteniendo su secuencia de entrada óptima $U_i^*(t)$ fija para encontrar su secuencia de entrada vecina deseada $U_j^{\mathrm{w}_i}(t)$. Aquí, también se considera que los subsistemas *l* siguen sus trayectorias actuales.

$$U_j^{\mathrm{w}_i}(t) = \arg\min_{U_j(t)} J_i\big(x_i(t), U_i^*(t), U_j(t), U_l^\mathrm{s}(t)\big), \tag{10}$$

5. Comunicación entre agentes: el agente *i* envía $U_j^{\mathrm{w}_i}(t)$ al agente *j* y recibe $U_i^{\mathrm{w}_j}(t)$.

3.2. Negociaciones difusas por pares

Al algoritmo DMPC le sigue la negociación difusa (Masero et al., 2021) para obtener una solución de control que garantice la estabilidad y disminuya el índice de desempeño. La negociación se realiza entre cada pareja de agentes $\{U_i^*(t), U_i^\mathrm{s}(t), U_i^{\mathrm{w}_j}(t)\}$ y $\{U_j^*(t), U_j^\mathrm{s}(t), U_j^{\mathrm{w}_i}(t)\}$. El sistema de inferencia difusa para la negociación genera la acción de control final $U_i^\mathrm{f}(t)$ y $U_j^\mathrm{f}(t)$ que llega a la segunda capa, teniendo en cuenta algunas restricciones operacionales y económicas.

3.3. Negociación RL por pares

El método propuesto se implementa en esta capa de la arquitectura de control, dado el agente *i* acoplado a través de entradas con sus vecinos y las dos de secuencias de control $\left\{U_i^{f_1}(t), U_i^{f_2}(t)\right\}$ obtenidas de dos negociaciones difusas por pares, esta última capa es la encargada de aplicar el proceso de negociación por pares empleando una red neuronal profunda bajo el método de aprendizaje por refuerzo.

La secuencia de control final $U_i^f(t)$ se calcula como la ponderación de las dos secuencias de control de las negociaciones difusas por pares de la capa anterior:

$$U_i^f(t) = U_i^{f_1}(t)\cdot\phi_i + U_i^{f_2}(t)\cdot(1-\phi_i), \tag{11}$$

donde ϕ_i es el valor discreto de la salida de la red neuronal que otorga un peso a cada secuencia de la negociación par del agente *i*, con $\phi_i \in [0\,1]$. El valor de ϕ_i permite que en la negociación la secuencia final $U_i^f(t)$ pueda ser $U_i^{f1}(t)$ o $U_i^{f2}(t)$ o la ponderación de ambas secuencias.

4. Algoritmo del aprendizaje por refuerzo

La red aprende la negociación a través del método de aprendizaje por refuerzo en (Sutton and Barto, 2018), esta negociación permite la optimización de cualquier índice de comportamiento dinámico relacionado con la estabilidad, el rendimiento, la economía y el tratamiento de restricciones adicionales no contempladas en los niveles inferiores del sistema jerárquico de control.

En el aprendizaje por refuerzo la política determina qué acción a_t tomar en función de un estado s_t dado por el entorno. Los agentes basados en políticas parametrizan directamente la función de la política $\pi_\theta(\cdot)$ en referencia a los parámetros θ. El agente implentado es el de gradiente de política (PG) (Williams, 1992), que es un método de aprendizaje por refuerzo en línea y que no requiere del modelo. Un agente PG es un agente basado en políticas que utiliza el algoritmo REINFORCE para buscar una política óptima que maximice la recompensa acumulada esperada a largo plazo G_t.

La política utilizada por el agente PG implementado es una política estocástica discreta. La política π_θ estocástica genera una distribución categórica de probabilidad sobre las acciones $\pi_\theta(s_t|a_t) = \mathbb{P}[s_t|a_t]$, siendo $\mathbb{P}$ la función de probabilidad. Una distribución categórica es una distribución de probabilidad discreta que describe los posibles resultados de una variable aleatoria que puede tomar una de k, con $k>0$, categorías posibles. Con la probabilidad de cada categoría especificada por separado $p_1, ..., p_k$. Los parámetros que especifican las probabilidades de cada resultado posible están limitados a estár en el rango de 0 a

1, y todos deben sumar a 1, expresado como $p_i > 0, \sum p_i = 1$. Los agentes basados en políticas abordan la búsqueda de la política, Algoritmo 3, como un problema de optimización. En el agente PG la política es optimizada con el método de optimización de gradiente de políticas (comparte el nombre con el agente). Este método emplea el gradiente ascendente para encontrar los valores de los parámetros θ con los que se obtiene el máximo local de la función objetivo del agente PG, $\nabla_\theta ln\,\pi(s_t;\theta)\cdot G_t$.

Algoritmo 3: Algoritmo de entrenamiento: REINFORCE.

1. Se inicializa la política $\pi(s;\theta)$ con valores aleatorios de θ.
2. En cada episodio de entrenamiento, se genera la experiencia del episodio siguiendo la política $\pi(s)$. El agente aplica acciones hasta que se alcanza el estado terminal s_T. La experiencia del episodio consiste en una secuencia $(s_0, a_0, r_1, s_1, ..., s_{T-1}, a_{T-1}, r_T, s_T)$ donde, s_t es el estado, a_t es la acción aplicada dado un estado, s_{t+1} es el siguiente estado, y r_{t+1} es la recompensa recibida al pasar del estado s_t al s_{t+1}.
3. Para cada estado en la secuencia del episodio, con $t = 1, 2, ..., T-1$, se calcula recompensa acumulada esperada G_t, con $G_t = \sum_{c=t}^{T} \gamma^{c-t} r_c$.
4. Se calcula el gradiente para maximizar la recompensa acumulada esperada. $d\theta = \sum_{t=1}^{T-1} G_t \cdot \nabla_\theta ln\,\pi(s_t;\theta)$
5. Se actualizan los parámetros de la política aplicando el gradiente, $\theta = \theta + \alpha d\theta$
   Aquí, α es la tasa de aprendizaje de la política.
6. Se repite del punto 2 al 5 para cada episodio de entrenamiento hasta que el entrenamiento finaliza.

El REINFORCE se basa en trayectorias, donde el retorno de una trayectoria τ con $R(\tau)$ se calcula como la recompensa total de esa trayectoria: $R(\tau) = (G_t, G_{t+1}, ..., G_{T-1})$, donde cada G_t es la recompensa que se espera acumular desde el instante t hasta T-1. La función de puntuación $\nabla_\theta ln\,\pi(s_t;\theta)$ permite que se logre la optimización sin requerir del modelo dinámico del sistema, optimizando el gradiente del logaritmo de $\pi(s_t|a_t)$ que expresa la probabilidad de que el agente seleccione la acción a_t dado un estado s_t.

5. Caso de estudio

La planta de laboratorio sobre la que se han probado las estrategias de control planteadas en este trabajo está formada por ocho tanques acoplados figura (1). El sistema se divide en $N = 4$ subsistemas: los tanques 1 y 3 pertenecen al subsistema 1; los tanques 2 y 4 forman el subsistema 2; los tanques 5 y 7 forman parte del subsistema 3; y el resto tanques forman el subsistema 4. La planta es controlada por 4 bombas que proporcionan un caudal q_m con $m \in \{a, b, c, d\}$ ubicadas en los subsistemas 1, 2, 3 y 4 respectivamente.
Cada subsistema tiene asociado un agente. Cada agente es un controlador local DMPC que utiliza un modelo local en variables de estado y una función objetivo local, que calcula la señal para su subsistema y la señal para su subsistema vecino.

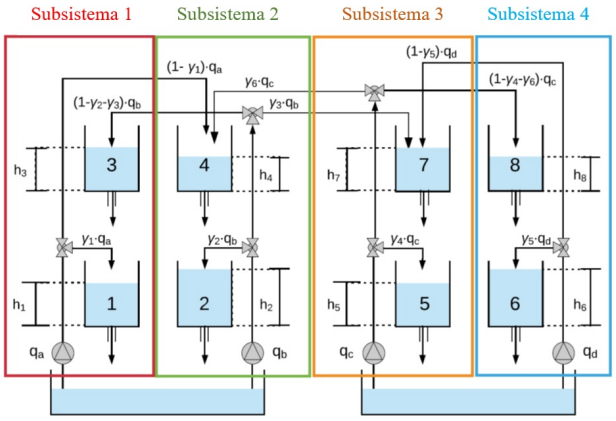

Figura 1: Sistema de 8 tanques interconectados.

h_n es el nivel del tanque $n \in \{1, 2, ..., 8\}$, se define el punto de operación del nivel del tanque h_n^0 para $n \in \{1, 2, ..., 8\}$ como $h_1^0 = 0{,}1$, $h_2^0 = 0{,}15$, $h_3^0 = 0{,}07$, $h_4^0 = 0{,}03$, $h_5^0 = 0{,}1$, $h_6^0 = 0{,}15$, $h_7^0 = 0{,}025$, $h_8^0 = 0{,}1$ (unidades: metros), y el punto de funcionamiento de tasa de flujo de las bombas q_m^0 para $m \in \{a, b, c, d\}$ como $q_a^0 = 0{,}142$, $q_b^0 = 0{,}421$, $q_c^0 = 0{,}421$, $q_d^0 = 0{,}140$ (unidades: m^3/h). El modelo de espacios de estado no lineal y de tiempo discreto es

$$\tilde{x}_N(t+1) = A_N \tilde{x}_N(t) + B_N \tilde{u}_N(t), \tag{12}$$

donde el vector de estados es $\tilde{x}_N = \left[h_1(t) - h_1^0, ..., h_8(t) - h_8^0\right]^\intercal$; el vector de entradas $\tilde{u}_N = \left[q_a(t) - q_a^0, ..., q_d(t) - q_d^0\right]^\intercal$; y A_N, B_N son las correspondientes matrices del sistema global. Similarmente, el sistema dinámico de cada subsistema $i \in \{1, 2, 3, 4\}$ son

$$\begin{aligned} \tilde{x}_i(t+1) &= A_i \tilde{x}_i(t) + B_{ii} \tilde{u}_i(t) + \tilde{w}_i(t), \\ \tilde{w}_i(t) &= \textstyle\sum_{j \in \mathcal{N}_i} B_{ij} \tilde{u}(t), \end{aligned} \tag{13}$$

donde $w_i(t)$ representa el acoplamiento con sus vecinos $j \in \mathcal{N}_i$, y las correspondientes matrices A_i B_{ii} B_{ij} de los subsistemas se pueden ver en (Masero et al., 2021). Además, los vectores de estados y entradas están limitados por

$$\begin{gathered} -h_n^0 < \tilde{x}_n(t) \leqslant 0{,}08, -q_m^0 < \tilde{u}_n(t) \leqslant 0{,}04, \\ \forall n \in \{1, 2, ..., 8\}, \forall m \in \{a, b, c, d\}. \end{gathered}$$

Las matrices $Q_\mathcal{N} = diag(Q_i)_{i \in \mathcal{N}}$ y $R_\mathcal{N} = diag(R_i)_{i \in \mathcal{N}}$ son, respectivamente, las correspondientes matrices de pesos constantes. Donde Q_i y R_i se pueden ver en (Masero et al., 2021). Además, la ganancia de retroalimentación local K_i y la matriz de ponderación del costo terminal P_i son definidas según Masero et al. (2021).
El objetivo de control es un problema de regulación en el que los niveles (h_1, h_2, h_5, h_6) deben alcanzar los niveles de referencia $(h_{r1}, h_{r2}, h_{r5}, h_{r6})$ considerando el costo operativo y satisfaciendo las restricciones operativas. Por lo tanto, es un problema de control multivariable con cuatro variables controladas (h_1, h_2, h_5, h_6) y cuatro variables manipuladas (q_a, q_b, q_c, q_d).

6. Elementos del agente PG en negociación entre agentes

Estos elementos del agente PG son el entorno, el estado s_t, la acción discreta a_t, la recompensa r_{t+1} y la red neuronal profunda empleada como función de aproximación de la política (ver

manual de MATLAB, Reinforcement Learning Toolbox). El entorno formado por los subsistemas y su arquitectura de control, recibe la acción, y consecuencia de esta, le envia al agente un estado y una recompensa, descritas en las siguientes secciones.

6.1. Estado y acción

El estado que envía el entorno al agente está formada por 20 valores que son las cuatro secuencias de señales de la capa inferior que se negocian por pares con $N_p = 5$.
La negociación se realiza entre las secuencias de señales del agente 2, $\left\{U_2^{f1}, U_2^{f2}\right\}$, y entre las señales del agente 3, $\left\{U_3^{f1}, U_3^{f2}\right\}$, siendo el estado $s_t = \left\{U_2^{f1\intercal}, U_2^{f2\intercal}, U_3^{f1\intercal}, U_3^{f2\intercal}\right\}$.
La acción $a_t = \{\phi_1, \phi_2\}$ que pertenecen al intervalo [0, 1]. Siendo $\phi_i = 0{,}05 \cdot \gamma$ con $i \in \{1, 2\}$ y $\gamma \in \{1, 2, 3, ..., 20\}$. Cada valor de $\phi_i n$ otorga un peso específico a las señales continuas para obtener la señales definitivas discretizadas, $U_2^f(k)$ y $U_3^f(k)$, de los subsistemas 2 y 3:

$$U_2^f(t) = U_2^{f1}(t) \cdot \phi_1 + U_2^{f2}(t) \cdot (1 - \phi_1) \tag{14}$$

$$U_3^f(t) = U_3^{f1}(t) \cdot \phi_2 + U_3^{f2}(t) \cdot (1 - \phi_2) \tag{15}$$

6.2. Recompensa

Durante el entrenamiento el entorno emplea la recompensa r_{t+1} para indicarle al agente la calificación que otorga el entorno a la acción a_t, según el objetivo del entorno. La recompensa es un escalar que para el correcto aprendizaje debe contener información lo suficientemente representativa del objetivo del entorno, siendo el objetivo del entorno ajustar los niveles de los tanques (h_1, h_2, h_5, h_6) a cada uno de sus niveles referencia objetivo $(h_{r1}, h_{r2}, h_{r5}, h_{r6})$.
La función de recompensa $r_{t+1}(\cdot)$, ecuación (16), está en función de la diferencia al cuadrado entre el nivel actual de los tanques 1, 2, 5 y 6, y sus niveles de referencia objetivo, formalizadas por la siguiente expresión, ecuaciones (17). con $n \in \{1, 2, 5, 6\}$

$$r_{t+1}(e_n) = \begin{cases} 2000, & Si\ e_n < e_{b_n}\ \forall n \\ -1000, & En\ caso\ contrario \end{cases} \tag{16}$$

$$e_n = (h_n(t) - h_{rn}(t))^2. \tag{17}$$

La acción a_t dado un estado s_t determina $h_n(t+1)$. Para que durante el entrenamiento el agente obtenga el máximo valor de recompensa , cada e_n debe ser menor que su correspondiente e_{b_n} con $n \in \{1, 2, 5, 6\}$, de lo contrario la recompensa será -1000. Los valores de e_{b_n} obtenidos de forma empírica son $e_{b_1} = 6{,}21 \cdot 10^{-8}$, $e_{b_2} = 1{,}5 \cdot 10^{-8}$, $e_{b_5} = 8{,}5 \cdot 10^{-9}$, $e_{b_6} = 1{,}8 \cdot 10^{-7}$.
Para evitar que la acción esté fuera de rango $[0, 1]$ se impone una penalización z_{t+1} que aumenta cuanto más lejos de este rango se encuentran los valores de la acción, como se muestra en la ecuación (18). Si $\phi_i \notin [0, 1]$ entonces $r_{t+1} = z_{t+1}$, donde $\phi_i = \{\phi_1, \phi_2\}$.

$$z_{t+1} = -3000 - ((|\phi_1| + |\phi_2|) \times 1000) \tag{18}$$

6.3. Red neuronal profunda

La red neuronal profunda, cuya estructura se muestra en la tabla 1, fue empleada como función de aproximación de la política. El algoritmo REINFORCE obtiene los pesos que configuran a la red para que devuelva una acción a_t que maximiza la recompensa esperada acumulada a largo plazo dado un estado s_t. La tasa de aprendizaje del agente , $\alpha = 0{,}001$, es un hiperparámetro que controla cuánto debe cambiar el modelo en respuesta al error estimado cada vez que se actualizan los pesos del modelo. Los canales de entrada de la red se corresponde con el estado, y la salida de la red es la acción con tantos elementos como el número de las posibles combinaciones de valores discretos.

Tabla 1: Red neuronal de la política.

Capas	Neuronas
FeatureInputLayer	20
FullyConnectedLayer	400
RelULayer	400
FullyConnectedLayer	400
RelULayer	400
FullyConnectedLayer	300
TanhLayer	300
ScalingLayer	300
FullyConnectedLayer	465

7. Resultados

7.1. Entrenamiento

Los resultados de la simulación se han obtenido utilizando la Reinforcement Learning Toolbox de MATLAB. El entrenamiento consistió en 1000 episodios de 100 pasos t cada episodio, Donde la recompensa total en cada episodio ep_s es la suma de las recompensas de 100 pasos, $ep_s = \sum_{t=1}^{100} r_t$ con $s = 1, 2, ..., 1000$. El periodo de muestreo utilizado por los controladores predictivos en los niveles más bajos fue $T = 5$. Las condiciones de entrenamiento fueron: $h_{r_n} = h_n^0 + 0{,}02 \cdot h_n^0$ y el punto de partida seleccionado de forma aleatoria al principio de cada episodio del intervalo siguiente $\left[h_n^0 - h_n^0 \cdot 0{,}01, h_n^0 + h_n^0 \cdot 0{,}01\right]$.

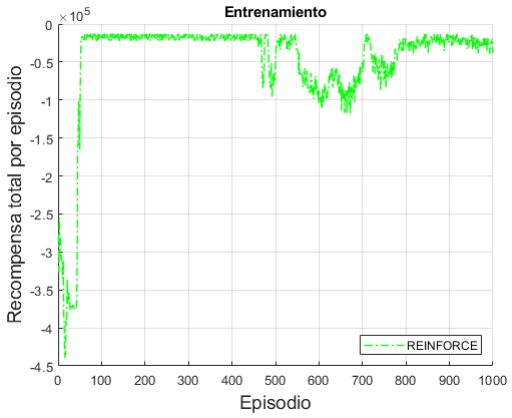

Figura 2: Entrenamiento agente PG.

El gráfico de entrenamiento figura 2. muestra la suma de las recompensas obtenidas en todos los pasos de cada episodio de

entrenamiento. La suma pequeña de los valores de las recompensas corresponde a una alta presencia de penalizaciones. A medida que la política converge hacia la política óptima estas penalizaciones son menos recurrentes, por lo que la suma de las recompensas tomará valores más altos.

7.2. Validación

Para validar los resultados se realizo una simulación de 100 segundos, con el mismo h_{r_n} utilizado en el entrenamiento y con h_n^0 como punto de partida. Los resultados del agente propuesto (PG DMPC) son comparados con el DMPC con lógica difusa (FL DMPC) (Masero et al., 2021) y el MPC centralizado, considerando los siguientes índices de desempeño:

- $Pred_{Total} = \sum_i J_i(\cdot)$, la suma de los errores de predicción de los tanques 1, 2, 5 y 6, más el esfuerzo de control de las bombas, con $i \in$ a todos los subsistemas $\{1, 2, 3, 4\}$.
- $ISE_{Total} = \sum_n ISE_n$, la suma de las integrales de los errores cuadráticos de los niveles de los tanques n, siendo cada $ISE_n = \int_0^{100} (h_n(t) - h_{rn}(t))^2 \, dt$.
- $Er_{Total}(t) = \sum_n (h_n(t) - h_{rn}(t))^2$, la suma de las diferencias al cuadrado de los niveles de los tanques n,
- $EB_{Total}(t) = \sum_m PE_m(t)$, suma de las energías de bombeo m, para cada bomba m, como la suma del promedio de la energía de bombeo sobre el horizonte de predicción, siendo proporcional a los flujos de agua proporcionadas por las bombas:

$$EB_m(t) = \frac{0{,}04}{3600 N_p} \sum_{k=1}^{N_p} q_m(t+k) \tag{19}$$

Como muestra la tabla 2. el control centralizado MPC es siempre el controlador con los valores de $Predict_{Total}$, ISE_{Total}, Er_{Total} y EB_{Total} más pequeños debido a la disponibilidad de información completa de la planta para las predicciones dado que utiliza un modelo global de la planta. Por otro lado, los controladores distribuidos (PG DMPC y FL DMPC) que utilizan los mismos modelos locales proporcionan, como cabía esperar, resultados parecidos.

Tabla 2: Comparación entre las técnicas consideradas

	PG DMPC	FL DMPC	Centralizado MPC
$Pred_{Total}$	0.032750	0.032615	0.019584
ISE_{Total}	0.043036	0.042962	0.030126
Er_{Total}	0.009407	0.009392	0.006825
EB_{Total}	0.006466	0.006467	0.006462

El PG DMPC emplea como criterio en la negociación sólo el error de ajuste de los niveles en forma de recompensa, mientras que el FL DMPC emplea más criterios, reglas difusas, dando al control mayor flexibilidad para un ajuste más preciso. No obstante, el algoritmo propuesto permite a futuro usar otros criterios a través de la recompensa, y además, a pesar de utilizar valores discretos presenta un buen ajuste de los niveles.

8. Conclusiones

Los resultados del PG DMPC tienen calidad suficiente para considerar al método una alternativa con coste computacional menor al control centralizado, y además, PG DMPC no requiere en su implementación de la complejidad que va implícita en un control centralizado. La metodología propuesta podría ser implementada en todas las capas de la arquitectura de control sin tener que formular los modelos locales del sistema, necesarios en un control centralizado que involucra mayor coste computacional.

9. Agradecimientos

Este trabajo ha sido realizado gracias al apoyo del proyecto PID2019-105434RB-C31 del programa Estatal de proyectos de investigación y del proyecto de la Fundación Samuel Solórzano FS/11-2021.

Referencias

Aponte, O., Vega, P., Francisco, M., 2022. Avances en informática y automática. decimoquinto workshop.

Cembellín, A., Francisco, M., Vega, P. I., 2019. Control predictivo centralizado y distribuido de una red de recogida de aguas residuales. In: XL Jornadas de Automática. Universidade da Coruña, Servizo de Publicacións, pp. 178–185.

Francisco, M., Mezquita, Y., Revollar, S., Vega, P., De Paz, J. F., 2019. Multi-agent distributed model predictive control with fuzzy negotiation. Expert Systems with Applications 129, 68–83.

Francisco, M., Skogestad, S., Vega, P., 2015. Model predictive control for the self-optimized operation in wastewater treatment plants: Analysis of dynamic issues. Computers & Chemical Engineering 82, 259–272.

Maestre, J. M., Muñoz, D., Camacho, E. F., 2011. Distributed model predictive control based on a cooperative game. Optimal Control Applications and Methods 32 (2), 153–176.

Masero, E., Francisco, M., Maestre, J., Revollar, S., Vega, P., 2021. Hierarchical distributed model predictive control based on fuzzy negotiation. Expert Systems with Applications, 176 (114836), 1-13.

Sierra-Garcia, J. E., Santos, M., Pandit, R., 2022. Wind turbine pitch reinforcement learning control improved by pid regulator and learning observer. Engineering Applications of Artificial Intelligence 111, 104769.

Sutton, R. S., Barto, A. G., 2018. Reinforcement learning: An introduction. MIT press.

Williams, R. J., 1992. Simple statistical gradient-following algorithms for connectionist reinforcement learning. Machine learning 8 (3), 229–256.