UNCUYO
UNIVERSIDAD NACIONAL DE CUYO

FACULTAD DE INGENIERÍA

INTELIGENCIA ARTIFICIAL II
UNIDAD 2: RAZONAMIENTO

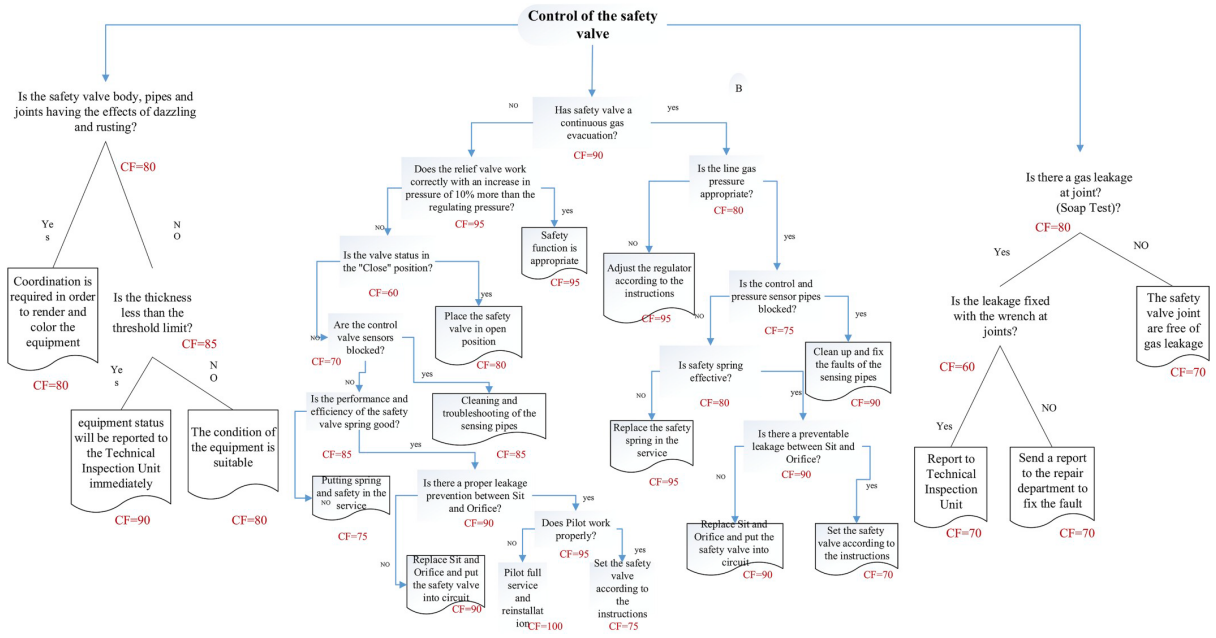
Grupo Nº1

Alumnos:

- ***Eula, Adriano German (L:12336)***
- ***Reinoso, Maximiliano Gabriel (L:11754)***
- ***Sena, Julieta (L:11367)***

Ejercicio 1: Knowledge-based system

Por medio de PROLOG se ha desarrollado una base de conocimientos para el sistema de evaluación y mantenimiento de válvulas de seguridad en estaciones reductoras de presión de gas. Para ello se utilizó el siguiente esquema simplificado de toma de decisiones:



A partir de esto, se creó un algoritmo de búsqueda en profundidad hacia atrás basado en los distintos estados del sistema. Dichos estados fueron agregados como 'Ground Facts' de manera aleatoria para simular diversas posibles situaciones.

Los componentes usados son los siguientes:

thickness_less_than_the_threshold_limit : ¿El espesor de la válvula es menor al umbral mínimo?

effects_of_dazzling_and_rusting : ¿Hay efectos de deslumbre u oxidación en la válvula, las tuberías o juntutras del sistema?

piloto : ¿El piloto funciona correctamente?

leakage_prevention_between_sit_and_orifice : ¿Existe una correcta fuga de prevención entre asiento y orificio?

safety_valve_spring : ¿El funcionamiento y la eficiencia del resorte de la válvula de seguridad es bueno?

control_valve_sensors_blocked : ¿Están bloqueados los sensores de la válvula de control?

valve_status_closed : ¿El estado de la válvula está "Cerrada"?

relief_valve_work_with_10_percent_more_pressure : ¿La válvula de alivio funciona correctamente al aumentar un 10% la presión sobre la presión nominal de regulación?

preventable_leakage_between_sit_and_orifice : ¿Hay una fuga prevenible entre asiento y orificio?

safety_valve_has_continuous_gas_evacuation : ¿Hay una evacuación continua de gas en la válvula de seguridad?

safety_spring_effective : ¿Es eficaz el resorte de seguridad?

control_and_pressure_sensor_pipes_blocked : ¿Las tuberías de control y sensado de presión están bloqueadas?

apropiate_line_gas_pressure : ¿Es correcta la presión de la línea de gas?

gas_leakage_at_joint : ¿Hay una fuga de gas en las junturas (prueba con jabón)?

leakage_fixed_with_the_wrench_at_joints : ¿Está fijada la fuga de gas en las junturas, con una llave?

Ejercicio 2: Artificial Intelligence Planning

En este problema utilizamos el planificador Fast Downward para poder modelar a través de PDDL el dominio de transporte aéreo de cargas y también el dominio de un proceso a modo de ejemplo de CAPP(Computer-Aided Process Planning)

Sistema de transporte aéreo:

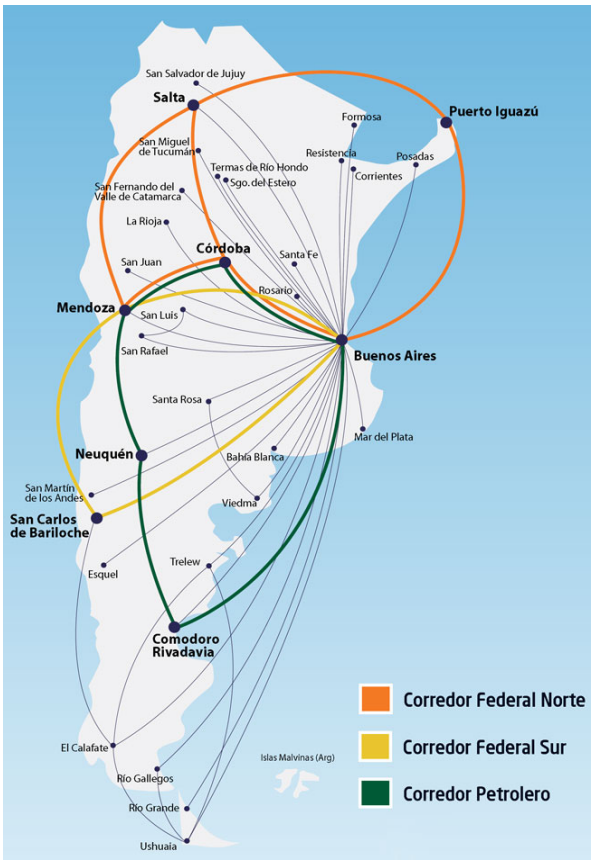


Imagen ilustrativa de las distintas rutas aéreas nacionales desde Ezeiza

En este problema tomamos como base el archivo ejemplo, con la diferencia que utilizamos 16 cargas ubicadas en 7 aeropuertos diferentes. Optamos por simular un servicio de entregas de encomiendas a nivel nacional y que la empresa sea Aerolíneas Argentinas. Los aeropuertos son los de:

Mendoza(MEN),Salta,Rosario(ROS),Misiones(MIS),Ezeiza,Ushuaia(UHA) y Santa Fe (STF). Las cargas no tienen denominación específica, las tomamos como cargas genéricas, sin ningún valor en especial por tipo de cargamento.

Al correr el código podemos observar que no hay interferencias con las instancias y las restricciones definidas y la velocidad de solución es bastante alta, no requiere para este tipo de ejemplo un costo computacional muy alto. Una variante que no probamos en esta simulación pero que consideramos necesaria para una aplicación real, serían que en el dominio podamos admitir un costo por cada carga dado su tamaño o valor y también tener en

cuenta los horarios que pista liberada de cada avión para poder luego lograr restringir el tiempo de entrega con el servicio de entregas de encomiendas. Otra opción sería agregar más aviones pero esto llevaría a tener que ampliar aún más la cantidad de nodos a explorar por el algoritmo y el costo computacional sería muy elevado teniendo en cuenta este ejemplo. Pero cabe aclarar que en una situación real se cuentan con los medios necesarios para llevar a cabo esta clase de planificaciones con flota de aviones aún más grandes y el costo computacional se ve amortizado por el servicio.

Resultados dados por el planner:

LOG	*PLAN*
Parsing: [0.010s CPU, 0.005s wall-clock] Normalizing task... [0.000s CPU, 0.000s wall-clock] Instantiating... Generating Datalog program... [0.000s CPU, 0.001s wall-clock] Normalizing Datalog program... Normalizing Datalog program: [0.000s CPU, 0.003s wall-clock] Preparing model... [0.000s CPU, 0.002s wall-clock] Generated 24 rules. Computing model... [0.020s CPU, 0.026s wall-clock] 481 relevant atoms 466 auxiliary atoms 947 final queue length 1102 total queue pushes Completing instantiation... [0.020s CPU, 0.019s wall-clock] Instantiating: [0.040s CPU, 0.051s wall-clock] Computing fact groups... Finding invariants... 3 initial candidates Finding invariants: [0.000s CPU, 0.002s wall-clock] Checking invariant weight... [0.000s CPU, 0.000s wall-clock] Instantiating groups... [0.010s CPU, 0.001s wall-clock] Collecting mutex groups... [0.000s CPU, 0.000s wall-clock] Choosing groups... 0 uncovered facts Choosing groups: [0.000s CPU, 0.001s wall-clock] Building translation key... [0.000s CPU, 0.000s wall-clock] Computing fact groups: [0.010s CPU, 0.006s wall-clock] Building STRIPS to SAS dictionary... [0.000s CPU, 0.000s wall-clock] Building dictionary for full mutex groups... [0.000s CPU, 0.000s wall-clock] Building mutex information... Building mutex information: [0.000s CPU, 0.000s wall-clock] Translating task... Processing axioms... Simplifying axioms... [0.000s CPU, 0.000s wall-clock] Processing axioms: [0.000s CPU, 0.001s wall-clock] Translating task: [0.020s CPU, 0.016s wall-clock] 0 effect conditions simplified 0 implied preconditions added Detecting unreachable propositions... 0 operators removed	(volar airbus380-315 ezeiza stf) (cargar carga11 airbus380-315 stf) (cargar carga5 airbus380-315 stf) (volar airbus380-315 stf mis) (cargar carga13 airbus380-315 mis) (cargar carga14 airbus380-315 mis) (cargar carga6 airbus380-315 mis) (volar airbus380-315 mis ezeiza) (descargar carga5 airbus380-315 ezeiza) (descargar carga6 airbus380-315 ezeiza) (volar airbus380-315 ezeiza men) (cargar carga1 airbus380-315 men) (descargar carga13 airbus380-315 men) (descargar carga14 airbus380-315 men) (cargar carga2 airbus380-315 men) (cargar carga8 airbus380-315 men) (volar airbus380-315 men ros) (descargar carga1 airbus380-315 ros) (cargar carga10 airbus380-315 ros) (descargar carga11 airbus380-315 ros) (descargar carga2 airbus380-315 ros) (cargar carga9 airbus380-315 ros) (volar airbus380-315 ros uha) (descargar carga10 airbus380-315 uha) (cargar carga15 airbus380-315 uha) (cargar carga16 airbus380-315 uha) (cargar carga3 airbus380-315 uha) (descargar carga9 airbus380-315 uha) (volar airbus380-315 uha salta) (cargar carga12 airbus380-315 salta) (descargar carga16 airbus380-315 salta) (cargar carga4 airbus380-315 salta) (descargar carga8 airbus380-315 salta) (volar airbus380-315 salta mis) (descargar carga12 airbus380-315 mis) (descargar carga15 airbus380-315 mis) (volar airbus380-315 mis ros) (descargar carga3 airbus380-315 ros) (descargar carga4 airbus380-315 ros)

17 propositions removed Detecting unreachable propositions: [0.000s CPU, 0.003s wall-clock] Translator variables: 17 Translator derived variables: 0 Translator facts: 135 Translator goal facts: 16 Translator mutex groups: 17 Translator total mutex groups size: 135 Translator operators: 266 Translator axioms: 0 Translator task size: 1325 Translator peak memory: 25508 KB Writing output... [0.010s CPU, 0.003s wall-clock] Done! [0.090s CPU, 0.086s wall-clock] Building causal graph... The causal graph is acyclic. 17 variables of 17 necessary 0 of 17 mutex groups necessary. 266 of 266 operators necessary. 0 of 0 axiom rules necessary. Building domain transition graphs... solveable in poly time 0 Building successor generator... Preprocessor facts: 135 Preprocessor derived variables: 0 Preprocessor task size: 1190 Writing output... done reading input... [t=0s] Simplifying transitions... done! done reading input! [t=0s] building causal graph...done! [t=0s] packing state variables...done! [t=0s] Variables: 17 Facts: 135 Bytes per state: 8 done initalizing global data [t=0s] Conducting best first search with reopening closed nodes, (real) bound = 2147483647 Initializing FF heuristic... Initializing additive heuristic... Simplifying 266 unary operators... done! [266 unary operators] f = 36 [1 evaluated, 0 expanded, t=0s, 3204 KB] Best heuristic value: 36 [g=0, 1 evaluated, 0 expanded, t=0s, 3204 KB] f = 37 [8 evaluated, 1 expanded, t=0s, 3204 KB] Best heuristic value: 35 [g=2, 9 evaluated, 2 expanded, t=0s, 3204 KB] Best heuristic value: 34 [g=3, 18 evaluated, 3 expanded, t=0s, 3204 KB] Best heuristic value: 33 [g=4, 26 evaluated, 4 expanded, t=0s, 3204 KB] Best heuristic value: 32 [g=5, 165 evaluated, 29 expanded, t=0s, 3204 KB] Best heuristic value: 31 [g=6, 187 evaluated, 32 expanded, t=0s, 3204 KB] Best heuristic value: 30 [g=7, 198 evaluated, 33 expanded, t=0s, 3204 KB] f = 38 [510 evaluated, 75 expanded, t=0s, 3204 KB] Best heuristic value: 29 [g=9, 514 evaluated, 76 expanded, t=0s, 3204 KB] Best heuristic value: 28 [g=10, 544 evaluated, 79 expanded, t=0s, 3204 KB]	; cost = 39 (unit cost)

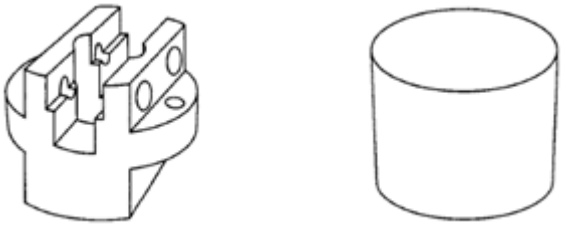
Best heuristic value: 27 [g=11, 559 evaluated, 80 expanded, t=0s, 3204 KB]
Best heuristic value: 26 [g=12, 654 evaluated, 89 expanded, t=0s, 3204 KB]
Best heuristic value: 25 [g=13, 668 evaluated, 90 expanded, t=0s, 3204 KB]
Best heuristic value: 24 [g=14, 1119 evaluated, 139 expanded, t=0s, 3204 KB]
Best heuristic value: 23 [g=15, 1130 evaluated, 140 expanded, t=0s, 3204 KB]
Best heuristic value: 22 [g=16, 1141 evaluated, 141 expanded, t=0s, 3204 KB]
Best heuristic value: 21 [g=17, 1143 evaluated, 142 expanded, t=0s, 3204 KB]
Best heuristic value: 20 [g=18, 1156 evaluated, 144 expanded, t=0s, 3204 KB]
Best heuristic value: 19 [g=19, 1168 evaluated, 145 expanded, t=0s, 3204 KB]
Best heuristic value: 18 [g=20, 1179 evaluated, 146 expanded, t=0s, 3204 KB]
Best heuristic value: 17 [g=21, 1190 evaluated, 147 expanded, t=0s, 3204 KB]
Best heuristic value: 16 [g=22, 1202 evaluated, 148 expanded, t=0.02s, 3204 KB]
f = 39 [19334 evaluated, 3073 expanded, t=0.16s, 4180 KB]
Best heuristic value: 15 [g=24, 19339 evaluated, 3075 expanded, t=0.16s, 4180 KB]
Best heuristic value: 14 [g=25, 19350 evaluated, 3076 expanded, t=0.16s, 4180 KB]
Best heuristic value: 13 [g=26, 19360 evaluated, 3077 expanded, t=0.16s, 4180 KB]
Best heuristic value: 12 [g=27, 19437 evaluated, 3086 expanded, t=0.16s, 4180 KB]
Best heuristic value: 11 [g=28, 19449 evaluated, 3087 expanded, t=0.16s, 4180 KB]
Best heuristic value: 10 [g=29, 19452 evaluated, 3088 expanded, t=0.16s, 4180 KB]
Best heuristic value: 9 [g=30, 19460 evaluated, 3089 expanded, t=0.16s, 4180 KB]
Best heuristic value: 8 [g=31, 19505 evaluated, 3094 expanded, t=0.16s, 4180 KB]
Best heuristic value: 7 [g=32, 19517 evaluated, 3095 expanded, t=0.16s, 4180 KB]
Best heuristic value: 6 [g=33, 19528 evaluated, 3096 expanded, t=0.16s, 4180 KB]
Best heuristic value: 5 [g=34, 19531 evaluated, 3097 expanded, t=0.16s, 4180 KB]
Best heuristic value: 4 [g=35, 19539 evaluated, 3098 expanded, t=0.16s, 4180 KB]
Best heuristic value: 3 [g=36, 19549 evaluated, 3099 expanded, t=0.16s, 4180 KB]
Best heuristic value: 2 [g=37, 19554 evaluated, 3100 expanded, t=0.16s, 4180 KB]
Best heuristic value: 1 [g=38, 19563 evaluated, 3101 expanded, t=0.16s, 4180 KB]
Best heuristic value: 0 [g=39, 19574 evaluated, 3102 expanded, t=0.16s, 4180 KB]
Solution found!
Actual search time: 0.16s [t=0.16s]
volar airbus380-315 ezeiza stf (1)
cargar carga11 airbus380-315 stf (1)
cargar carga5 airbus380-315 stf (1)
volar airbus380-315 stf mis (1)
cargar carga13 airbus380-315 mis (1)

```
cargar carga14 airbus380-315 mis (1)
cargar carga6 airbus380-315 mis (1)
volar airbus380-315 mis ezeiza (1)
descargar carga5 airbus380-315 ezeiza (1)
descargar carga6 airbus380-315 ezeiza (1)
volar airbus380-315 ezeiza men (1)
cargar carga1 airbus380-315 men (1)
descargar carga13 airbus380-315 men (1)
descargar carga14 airbus380-315 men (1)
cargar carga2 airbus380-315 men (1)
cargar carga8 airbus380-315 men (1)
volar airbus380-315 men ros (1)
descargar carga1 airbus380-315 ros (1)
cargar carga10 airbus380-315 ros (1)
descargar carga11 airbus380-315 ros (1)
descargar carga2 airbus380-315 ros (1)
cargar carga9 airbus380-315 ros (1)
volar airbus380-315 ros uha (1)
descargar carga10 airbus380-315 uha (1)
cargar carga15 airbus380-315 uha (1)
cargar carga16 airbus380-315 uha (1)
cargar carga3 airbus380-315 uha (1)
descargar carga9 airbus380-315 uha (1)
volar airbus380-315 uha salta (1)
cargar carga12 airbus380-315 salta (1)
descargar carga16 airbus380-315 salta (1)
cargar carga4 airbus380-315 salta (1)
descargar carga8 airbus380-315 salta (1)
volar airbus380-315 salta mis (1)
descargar carga12 airbus380-315 mis (1)
descargar carga15 airbus380-315 mis (1)
volar airbus380-315 mis ros (1)
descargar carga3 airbus380-315 ros (1)
descargar carga4 airbus380-315 ros (1)
Plan length: 39 step(s).
Plan cost: 39
Initial state h value: 36.
Expanded 3103 state(s).
Reopened 0 state(s).
Evaluated 19574 state(s).
Evaluations: 19574
Generated 35014 state(s).
Dead ends: 0 state(s).
Expanded until last jump: 3073 state(s).
Reopened until last jump: 0 state(s).
Evaluated until last jump: 19334 state(s).
Generated until last jump: 34687 state(s).
Number of registered states: 19574
Search time: 0.16s
Total time: 0.16s
Solution found.
Peak memory: 4180 KB
```

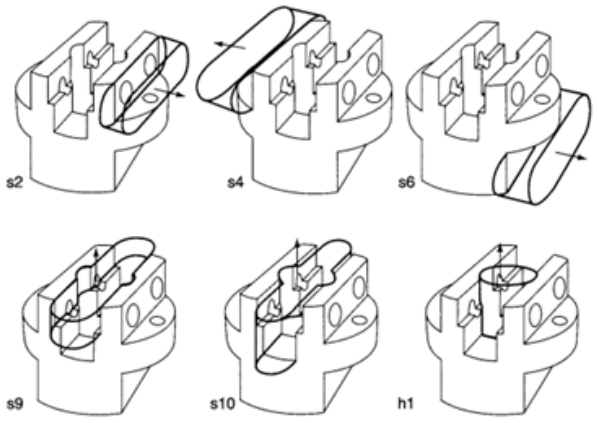
Podemos observar que para esta ocasión el costo del plan es de 39 correlacionado con la cantidad de steps completados y la cantidad de nodos expandidos fue de 3103.

Planificación de Procesos Asistida por Computadora

En este ejercicio de CAPP se utilizó el modelo visto en los ejemplos de clase.



En este caso utilizamos como instancia en el dominio la de la acción del fresado, ya que si bien es la utilizada en el ejemplo. También es la más representativa para poder llevar a cabo todo el mecanizado de la pieza. Inclusive se pueden reemplazar las tareas de taladrado para los agujeros pasantes y el agujero central ciego, por el mismo fresado. Pero para ese caso y llevarlo más a lo que sucede en un centro de mecanizado. Es permitirle a nuestro dominio el poder instanciar los momentos en los cuales la fresa cambia de herramienta. De esa manera en el mismo planner utilizamos una sola acción para lograr los distintos features de la pieza y no desperdiciamos costo computacional para una tarea que puede llevarse a cabo en una misma máquina. Esto lo observamos porque si deseamos continuar con el ejemplo y definir una instancia para además de fresado, agregar el de taladrado y torneado. El hecho de saber que son tres tareas diferentes que se realizan en máquinas diferentes. nos estaría obligando a tener que agregar un costo por cambio de máquina teniendo en cuenta los tiempos de mecanizado de cada máquina y teniendo en cuenta también los costos de puesta a punto de cada máquina para llevar las distintas tareas. En cambio si restringimos las diferentes tareas con sus respectivas herramientas pero sin cambiar de máquina. El código quedaría más compacto y simularía de una manera más óptima lo que sucede en un centro de mecanizado industrial.



Resultados dados por el planner:

LOG	***PLAN***
Parsing... Parsing: [0.000s CPU, 0.002s wall-clock] Normalizing task... [0.000s CPU, 0.000s wall-clock]	(op-fresado orientacion-x s6 slot fresado) (setup-orientacion orientacion-x orientacion+x) (op-fresado orientacion+x s9 slot fresado) (setup-orientacion orientacion+x orientacion+y)

Instantiating... Generating Datalog program... [0.000s CPU, 0.000s wall-clock] Normalizing Datalog program... Normalizing Datalog program: [0.010s CPU, 0.003s wall-clock] Preparing model... [0.000s CPU, 0.001s wall-clock] Generated 20 rules. Computing model... [0.000s CPU, 0.004s wall-clock] 109 relevant atoms 62 auxiliary atoms 171 final queue length 202 total queue pushes Completing instantiation... [0.010s CPU, 0.003s wall-clock] Instantiating: [0.020s CPU, 0.012s wall-clock] Computing fact groups... Finding invariants... 4 initial candidates Finding invariants: [0.000s CPU, 0.001s wall-clock] Checking invariant weight... [0.000s CPU, 0.000s wall-clock] Instantiating groups... [0.000s CPU, 0.000s wall-clock] Collecting mutex groups... [0.000s CPU, 0.000s wall-clock] Choosing groups... 5 uncovered facts Choosing groups: [0.000s CPU, 0.000s wall-clock] Building translation key... [0.000s CPU, 0.000s wall-clock] Computing fact groups: [0.000s CPU, 0.002s wall-clock] Building STRIPS to SAS dictionary... [0.000s CPU, 0.000s wall-clock] Building dictionary for full mutex groups... [0.000s CPU, 0.000s wall-clock] Building mutex information... Building mutex information: [0.000s CPU, 0.000s wall-clock] Translating task... Processing axioms... Simplifying axioms... [0.000s CPU, 0.000s wall-clock] Processing axioms: [0.000s CPU, 0.000s wall-clock] Translating task: [0.000s CPU, 0.002s wall-clock] 0 effect conditions simplified	(op-fresado orientacion+y s10 slot fresado) (op-fresado orientacion+y s2 slot fresado) (setup-orientacion orientacion+y orientacion-y) (op-fresado orientacion-y s4 slot fresado) ; cost = 8 (unit cost)

```
0 implied preconditions added
Detecting unreachable propositions...
0 operators removed
1 propositions removed
Detecting unreachable propositions: [0.000s
CPU, 0.001s wall-clock]
Translator variables: 6
Translator derived variables: 0
Translator facts: 16
Translator goal facts: 5
Translator mutex groups: 1
Translator total mutex groups size: 6
Translator operators: 35
Translator axioms: 0
Translator task size: 138
Translator peak memory: 24352 KB
Writing output... [0.000s CPU, 0.001s
wall-clock]
Done! [0.020s CPU, 0.019s wall-clock]
Building causal graph...
The causal graph is acyclic.
6 variables of 6 necessary
0 of 1 mutex groups necessary.
35 of 35 operators necessary.
0 of 0 axiom rules necessary.
Building domain transition graphs...
solveable in poly time 0
Building successor generator...
Preprocessor facts: 16
Preprocessor derived variables: 0
Preprocessor task size: 132
Writing output...
done
reading input... [t=0s]
Simplifying transitions... done!
done reading input! [t=0s]
building causal graph...done! [t=0s]
packing state variables...done! [t=0s]
Variables: 6
Facts: 16
Bytes per state: 4
done initalizing global data [t=0s]
Conducting best first search with reopening
closed nodes, (real) bound = 2147483647
Initializing FF heuristic...
Initializing additive heuristic...
Simplifying 35 unary operators... done! [35
unary operators]
f = 8 [1 evaluated, 0 expanded, t=0s, 2936 KB]
Best heuristic value: 8 [g=0, 1 evaluated, 0
expanded, t=0s, 2936 KB]
Best heuristic value: 7 [g=1, 2 evaluated, 1
expanded, t=0s, 2936 KB]
```

```
Best heuristic value: 6 [g=2, 8 evaluated, 2 expanded, t=0s, 2936 KB]
Best heuristic value: 5 [g=3, 13 evaluated, 3 expanded, t=0s, 2936 KB]
Best heuristic value: 4 [g=4, 14 evaluated, 4 expanded, t=0s, 2936 KB]
Best heuristic value: 3 [g=5, 19 evaluated, 5 expanded, t=0s, 2936 KB]
Best heuristic value: 2 [g=6, 21 evaluated, 6 expanded, t=0s, 2936 KB]
Best heuristic value: 1 [g=7, 30 evaluated, 7 expanded, t=0s, 2936 KB]
Best heuristic value: 0 [g=8, 32 evaluated, 8 expanded, t=0s, 2936 KB]
Solution found!
Actual search time: 0s [t=0s]
op-fresado orientacion-x s6 slot fresado (1)
setup-orientacion orientacion-x orientacion+x (1)
op-fresado orientacion+x s9 slot fresado (1)
setup-orientacion orientacion+x orientacion+y (1)
op-fresado orientacion+y s10 slot fresado (1)
op-fresado orientacion+y s2 slot fresado (1)
setup-orientacion orientacion+y orientacion-y (1)
op-fresado orientacion-y s4 slot fresado (1)
Plan length: 8 step(s).
Plan cost: 8
Initial state h value: 8.
Expanded 9 state(s).
Reopened 0 state(s).
Evaluated 32 state(s).
Evaluations: 32
Generated 51 state(s).
Dead ends: 0 state(s).
Expanded until last jump: 0 state(s).
Reopened until last jump: 0 state(s).
Evaluated until last jump: 1 state(s).
Generated until last jump: 0 state(s).
Number of registered states: 32
Search time: 0s
Total time: 0s
Solution found.
Peak memory: 2936 KB
```

En este caso, el costo del plan es de 8, uno por cada step realizado y se generaron 51 estados diferentes para llegar a la solución. Cabe destacar que este ejemplo es muy simplificado a lo que sucede en la vida real, ya que en ningún momento hemos instanciado las restricciones y el dominio para los distintos tipos de herramientas y la velocidad de avance de las mismas. También no se han agregado instancias para las terminaciones de rugosidad que en un centro de mecanizado tienen preponderancia.

Ejercicio 3: Lógica Difusa

Resolución del problema:

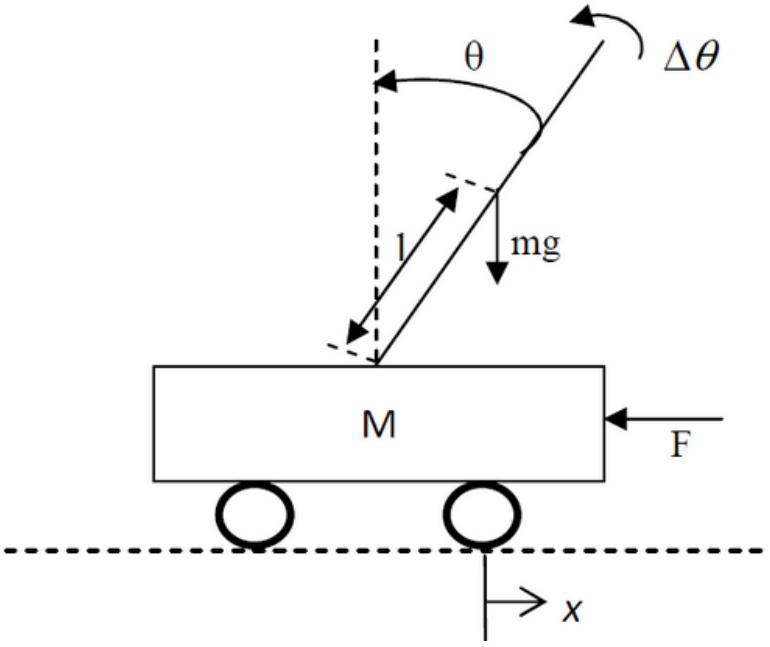
https://colab.research.google.com/drive/1TfuiUhv9CZCPgQwIQZyJ_EoxcUYhn2Qk#scrollTo=rchwkbAm_vuw

El objetivo del ejercicio es la implementación de un sistema de inferencia difusa para controlar un péndulo invertido montado sobre un carro que se mueve sin restricciones en sobre una dirección dada; considerando que el mismo no tiene un espacio restringido de movimiento, y que responde al siguiente modelo planteado:

$$\ddot{\theta} = \frac{g \sin\theta + \cos\theta \left(\frac{-F - ml\dot{\theta}^2 Sin\theta}{M+m} \right)}{l\left(\frac{4}{3} - \frac{m\cos^2\theta}{M+m} \right)}$$

$$\theta' = \theta' + \theta''\Delta t$$

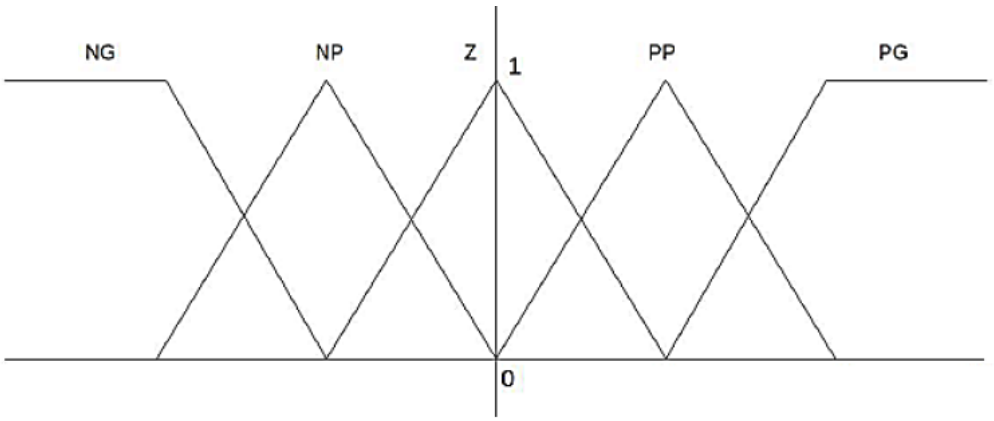
$$\theta = \theta + \theta'\Delta t + (\theta''\Delta t^2)/2$$



Las variables de entrada para nuestro problema son la posición inicial (tita_0) y velocidad angular inicial (v_tita_0). Nuestra salida es la Fuerza que hay que aplicar para alterar la posición del péndulo.

Se adoptaron para cada uno de los dominios 5 conjuntos borrosos NG, NP, Z, PP, PG. Las funciones de pertenencia adoptadas fueron en todos los casos triangulares y con hombro derecho e izquierdo en los extremos. Se decidió que el solapamiento de cada conjunto

borroso fuera el máximo, es decir del 50%. El borrosificador adoptado es del tipo "Singleton".



Habiendo definido cómo se vería nuestro sistema difuso procedimos a armar la FAM del mismo:

Variables lingüísticas de entrada:

tita_0 ={NG,NP,Z,PP,PG}

v_tita_0 ={NG,NP,Z,PP,PG}

Variable lingüística de salida:

fuerza ={NG,NP,Z,PP,PG}

v_tita_0 \ tita_0	NG	NP	Z	PP	PG
NG	PG	PG	PG	PP	NP
NP	PG	PG	PP	Z	NG
Z	PG	PP	Z	NP	NG
PP	PG	Z	NP	NG	NG
PG	PP	NP	NG	NG	NG

Dado que para el armado de la FAM se necesitaría el conocimiento experto, nosotros actuamos en ese rol usando nuestro criterio y ayudándonos de información obtenida de distintos trabajos que existen en internet sobre lógica difusa y péndulo invertido.

Con la FAM podemos elaborar 25 reglas de inferencia borrosa de Mamdani, las cuales son:

- R1: IF tita_0 is NG AND v_tita is NG THEN fuerza is PG

- R2: IF tita_0 is NG AND v_tita is NP THEN fuerza is PG
- R3: IF tita_0 is NG AND v_tita is Z THEN fuerza is PG
- R4: IF tita_0 is NG AND v_tita is PP THEN fuerza is PG
- R5: IF tita_0 is NG AND v_tita is PG THEN fuerza is PP

- R6: IF tita_0 is NP AND v_tita is NG THEN fuerza is PG
- R7: IF tita_0 is NP AND v_tita is NP THEN fuerza is PG
- R8: IF tita_0 is NP AND v_tita is Z THEN fuerza is PP
- R9: IF tita_0 is NP AND v_tita is PP THEN fuerza is Z
- R10: IF tita_0 is NP AND v_tita is PG THEN fuerza is NP

- R11: IF tita_0 is Z AND v_tita is NG THEN fuerza is PG
- R12: IF tita_0 is Z AND v_tita is NP THEN fuerza is PP
- R13: IF tita_0 is Z AND v_tita is Z THEN fuerza is Z
- R14: IF tita_0 is Z AND v_tita is PP THEN fuerza is NP
- R15: IF tita_0 is Z AND v_tita is PG THEN fuerza is NG

- R16: IF tita_0 is PP AND v_tita is NG THEN fuerza is PP
- R17: IF tita_0 is PP AND v_tita is NP THEN fuerza is Z
- R18: IF tita_0 is PP AND v_tita is Z THEN fuerza is NP
- R19: IF tita_0 is PP AND v_tita is PP THEN fuerza is NG
- R20: IF tita_0 is PP AND v_tita is PG THEN fuerza is NG

- R21: IF tita_0 is PG AND v_tita is NG THEN fuerza is NP
- R22: IF tita_0 is PG AND v_tita is NP THEN fuerza is NG
- R23: IF tita_0 is PG AND v_tita is Z THEN fuerza is NG
- R24: IF tita_0 is PG AND v_tita is PP THEN fuerza is NG
- R25: IF tita_0 is PG AND v_tita is PG THEN fuerza is NG

Por último se adoptó como método para desborrosificar la media por centro dado que es muy efectivo a un bajo costo computacional.

$$y = \frac{\sum_{l=1}^{l=M} \bar{y}^l \mu_{B'}(\bar{y}^l)}{\sum \mu_{B'}(\bar{y}^l)}$$

donde $\bar{y}^l$ es el punto máximo de B', y *M* es la cantidad de conjuntos borrosos de salida

Ya con toda esa información el procedimiento del algoritmo desarrollado es el siguiente.

Primero ingresamos los valores de tita_0 y v_tita_0 para ser borrosificados y obtener su valor de pertenencia, en nuestro caso dado que teníamos un solapamiento del 50% en muchos de los casos las variables iban a tener valor de pertenencia para dos conjuntos difusos. Esto nos implica que cada vez que ingresamos un valor de tita_0 y v_tita_0, íbamos a tener que aplicar de 1 a 4 de las 25 reglas. Un caso de una regla sería tener valor extremos en los hombros que nos darían valores de pertenencia a un solo conjunto borroso, tanto para tita_0 y v_tita_0.

En estas condiciones se puede dar que tengamos reglas con el mismo resultado o distinto resultado.

En caso de que dos reglas tuvieran el mismo resultado de fuerza se calcula el máximo valor de pertenencia entre ellas para obtener un solo resultado. En este caso, cuando calculemos la media de centros, nuestro valor resultante al desborrosificar sería el correspondiente al centro de ese conjunto.

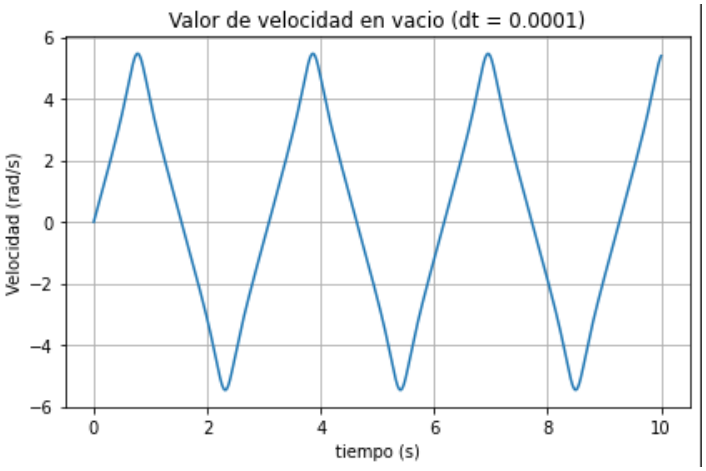
En caso de que tengamos reglas con distinto resultado, lo que se hace es la disyunción entre ambos conjuntos borrosos para obtener un solo conjunto borroso con los máximos de pertenencia de cada uno para luego aplicar la media de centros.

Estos dos casos nos indican que aplicando la lógica de 25 reglas se pueden agrupar todas las que tuvieran el mismo resultado para obtener solo 5 reglas. En nuestro caso el algoritmo trabaja con las 25 reglas y a medida que se recorren se aplica la lógica del caso de que haya reglas con el mismo resultado.

Ya sea cualquiera de los dos casos, con el valor de fuerza obtenido por media de centros se ingresa en la simulación del sistema fisico y se calculan los estados siguientes del mismo, luego se repite el procedimiento con los nuevos valores obtenidos de tita_0 y v_tita_0.

Resultados de nuestro algoritmo:

Para calcular el dominio de las velocidades se observó entre qué valores oscila el péndulo en vacío, dado estos valores se definió el dominio (-6,6).



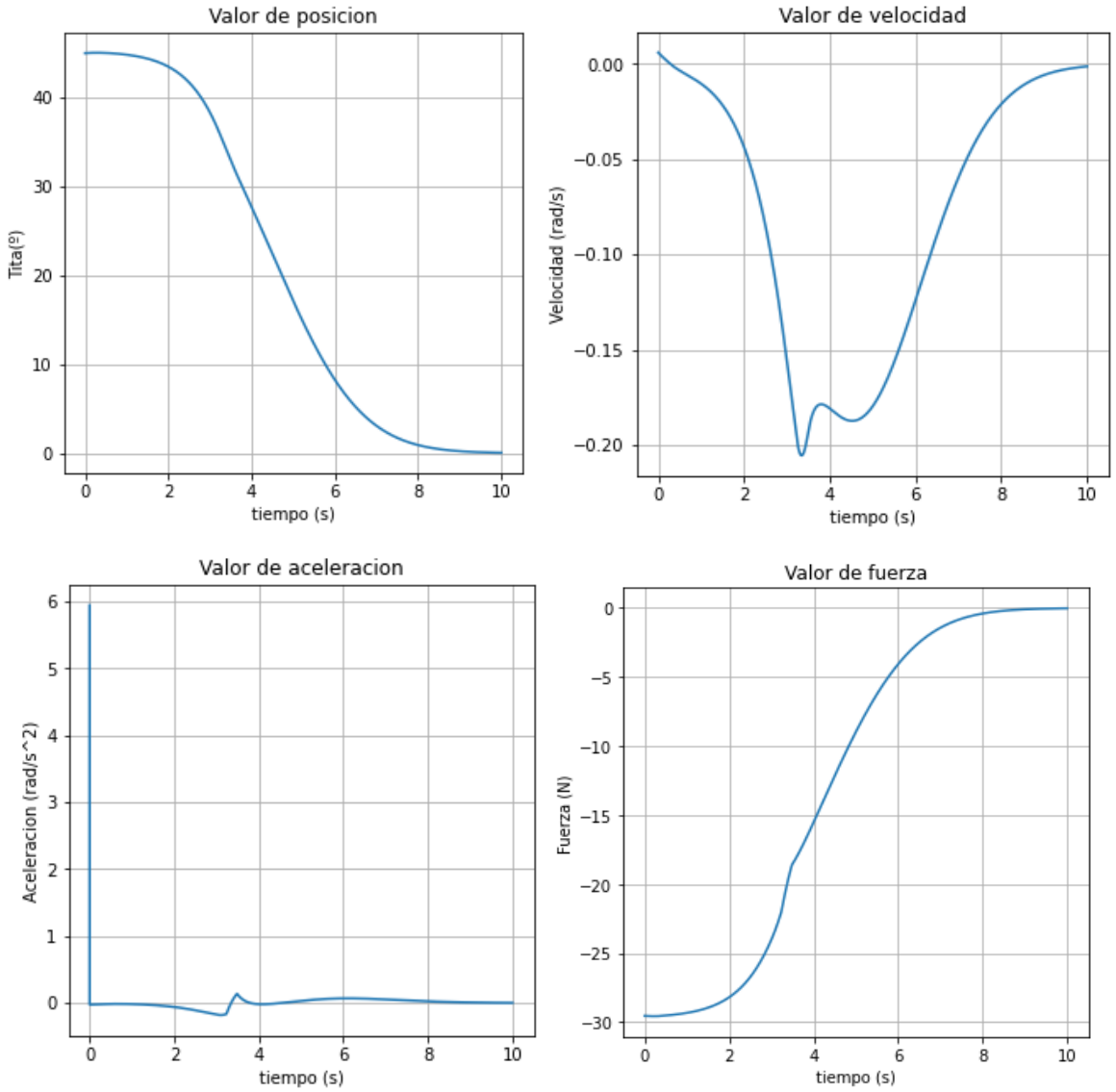
Para el caso de tita_0, siendo que nuestro ángulo 0º es la posición de equilibrio nuestro sistema, esto nos permite como máximo dar dominio entre (-180, 180), pero consideramos que el carro está apoyado en el suelo, pasar por debajo de la horizontal implicaría que la barra golpee el suelo. Por lo tanto, definimos el dominio de trabajo entre (-90,90).

Ahora presentaremos múltiples casos para determinar el funcionamiento de nuestro sistema.

Primero se observaron los casos de soltar el péndulo desde la posición de 45° sin velocidad inicial y se fueron variando el dominio de las fuerzas para ver tiempos de demora para llegar al equilibrio.

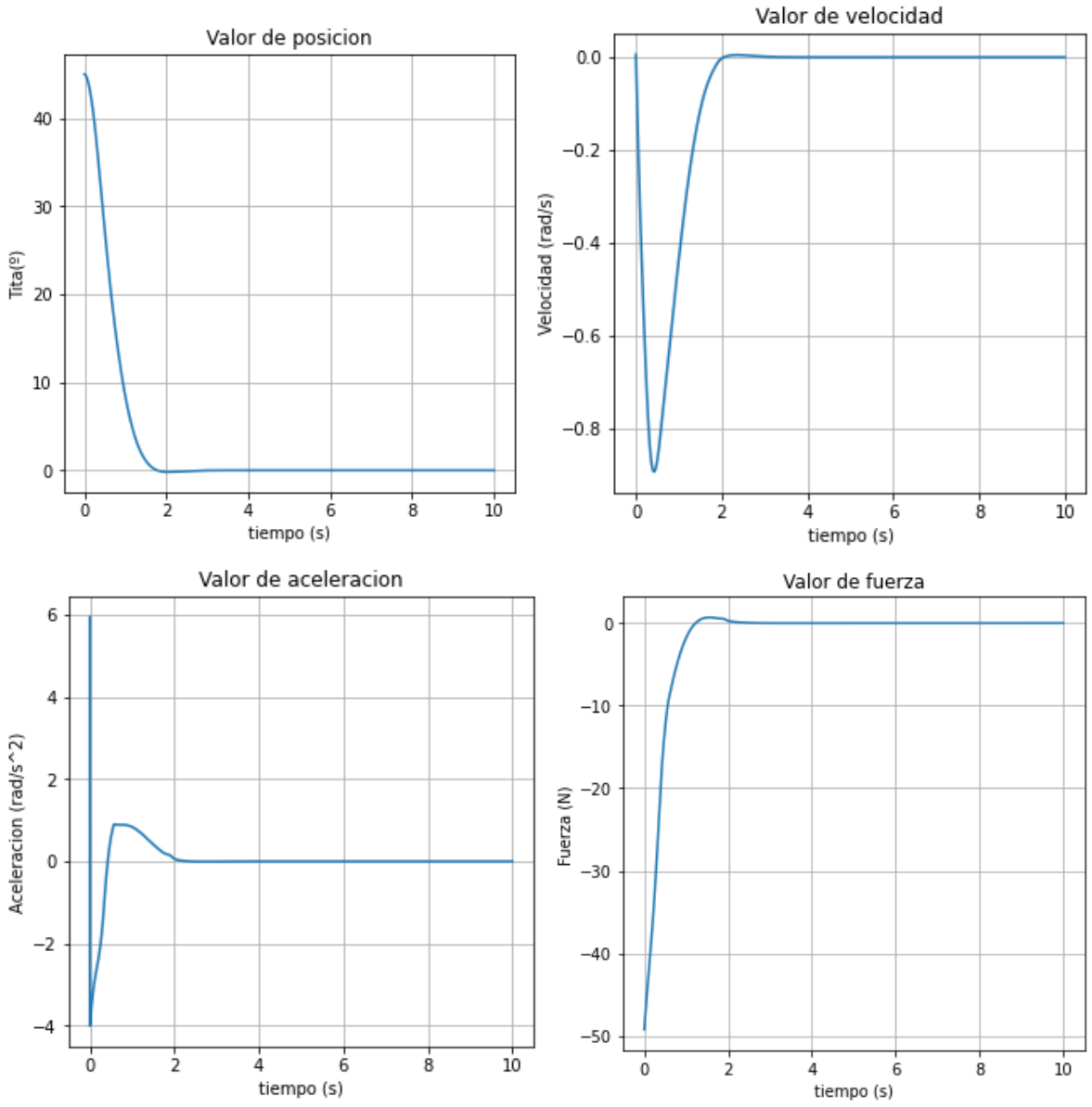
Primer caso:

Comenzamos con un dominio de fuerza pequeño (-60,60). Podemos ver que el sistema parte de su posición de 45° y tarda un tiempo considerable en estabilizarse. La fuerza llega a un valor máximo de -30 y el péndulo no presenta una aceleración considerable.



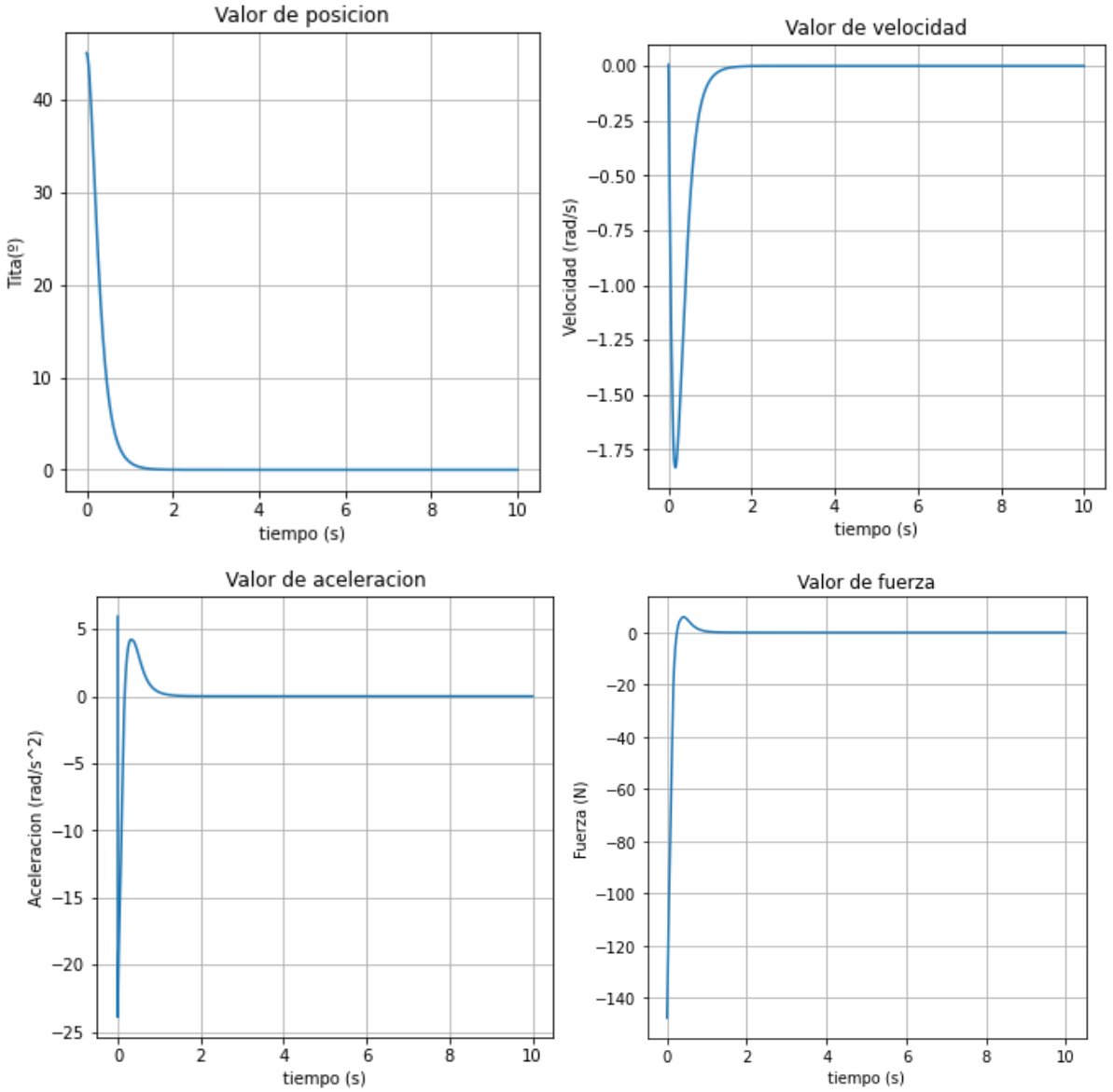
Segundo caso:

Para el caso de tener un dominio de fuerza (-100,100) el sistema se estabiliza después de 4 segundos y la fuerza aplicada máxima alcanza un valor de 50. En ningún momento el péndulo se aleja más de la posición de la cual fue soltado, por lo tanto, podemos aceptar que la solución es aplicable.



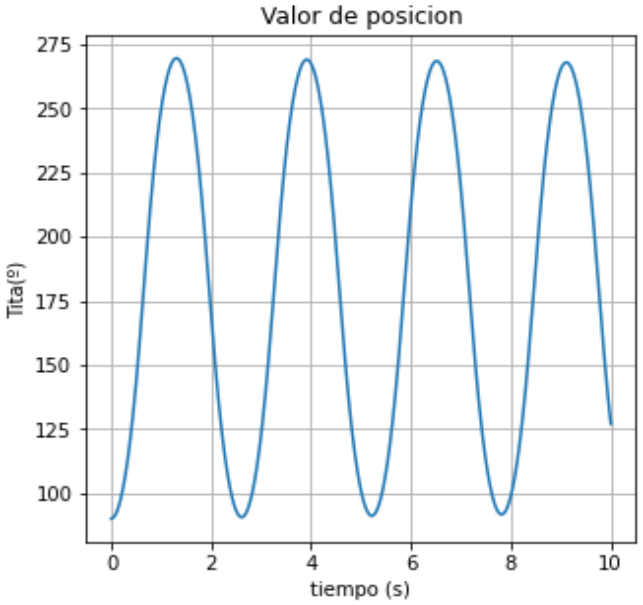
Tercer caso:

En este tenemos un dominio de fuerzas de (-300,300), la estabilización se logró en poco más de 2 segundos. Es un tiempo bastante bueno, pero estamos aplicando una fuerza de 150 y el péndulo experimenta una velocidad mayor.

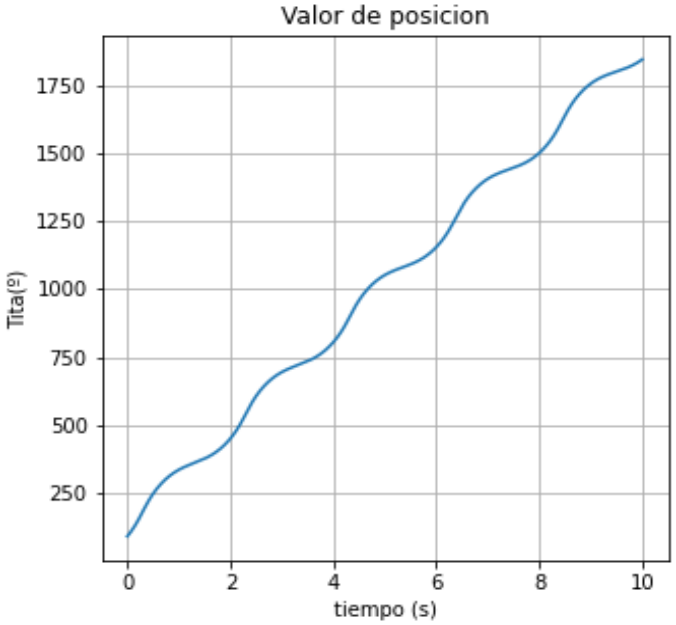


- Análisis del péndulo en el límite de nuestro dominio de posición (90°) con velocidad inicial nula:

Este comportamiento se debe a que al trabajar en el límite, nos salimos del dominio de trabajo de nuestro algoritmo de lógica difusa. Por lo cual es el equivalente a no haber hecho nada sobre el péndulo.



- Análisis del péndulo en el límite de nuestro dominio de posición (90º) con velocidad inicial distinta de cero:



Esta gráfica, para una velocidad inicial de 4 rad/s, nos indica que el sistema está en condiciones de trabajo que no puede actuar como deseamos. Se aplica una fuerza al comienzo intentando llevar al equilibrio pero se sale rápidamente del dominio y el péndulo comienza a girar en una sola dirección.

Para los casos en que damos una velocidad inicial distinta de 0 lo que se logra es que el sistema va a tener una perturbación extra, lo importante es que esta no saque al sistema fuera de los dominios de trabajo definidos de lo contrario no lograremos la convergencia del péndulo a 0.

Finalmente, del trabajo realizado determinamos:

La aplicación de lógica difusa para este caso no implicó un gran desarrollo de programación hoy en día incluso se podría simplificar en gran medida la programación usando Matlab que cuenta con librerías de lógica difusa.

La verdadera dificultad se encuentra en lograr analizar el problema real, saber elegir qué criterios de borrosificación aplicar, generar reglas de Mamdani que reflejen nuestro problema, en este punto se nota la importancia de trabajar con un experto o ver la necesidad de invertir un buen tiempo en la experimentación. Definir reglas adecuadas es la diferencia entre que nuestro problema se resuelva o no. Una vez definido todo es momento de elegir los dominios de cada una de nuestras variables, en este punto entran en juego las limitaciones de la vida real. Ver que costo implica cada resultado, los tiempos requeridos para estabilizar el sistema y en ciertos casos particulares, factores que mantengan la estabilidad de la estructura (no sobrepasar sus límites estructurales). El método para aplicar la fuerza puede limitarnos a ciertos valores y también esa aplicación de fuerza puede comprometer la estructura.

Por otro lado, este desarrollo no solo implica reconocer las buenas aplicaciones de lógica difusa, sino saber minimizar los tiempos de implementación con un buen armado de base de conocimiento, además de saber maximizar los resultados adaptándose a la disponibilidad de recursos en la realidad.