Relatório do Projeto 2

Projeto desenvolvido durante no 2º período do 2º semestre do ano letivo 2022/2023

1st Leonardo Amado Brito
Instituto Superior Técnico
Mestrado em Engenharia e Ciência de Dados
Lisboa, Portugal
leonardo.amado.brito@tecnico.ulisboa.pt

2nd Gustavo Caria
Instituto Superior Técnico
Mestrado em Matemática Aplicada e Computação
Lisboa, Portugal
gustavocaria@tecnico.ulisboa.pt

I. Introdução

No panorama digital em rápida evolução, a necessidade de um desempenho adaptativo e optimizado dos sistemas informáticos é fundamental. Este projeto visa responder a esta necessidade, tirando partido das capacidades dos Sistemas de Inferência Fuzzy (SIF), um tipo de Inteligência Artificial que se destaca no tratamento de dados incertos ou imprecisos. Ao interpretar parâmetros do sistema e da rede, como a utilização da memória, a carga do processador, a largura de banda e a latência, através de um SIF, pretendemos permitir um controlo mais flexível e adaptável dos sistemas informáticos.

O projeto envolverá a identificação dos principais parâmetros de desempenho, a conceção de regras fuzzy para descrever suas interações e a criação de um FIS para previsão e ajuste de desempenho em tempo real. A eficácia do SIF será avaliada utilizando um pequeno conjunto de dados de peritos, e será explorado o potencial dos métodos de conjunto para melhorar a robustez e a precisão.

II. Definir o problema

O objetivo do nosso sistema inteligente *fuzzy* é controlar a *CLP Variation* (a variação de carga computacional) dos dispositivos *edge* de uma rede de *IoT*, baseado nas variáveis:

- *Memory Usage* (*MU*)
- *Processor Load* (*PL*)
- *Input Network Throughput* (*InpNet*)
- *Output Network Throughput* (*OutNet*)
- *Available Output Bandwidth* (*Bandwidth*)
- *Latency*
- *CLP Variation* (*CLPV*)

Para tal, foi preciso definir o *range* do domínio de cada variável para o nosso problema. Inicialmente tínhamos algumas dúvidas quanto às variáveis cujas unidades não tinham sido especificadas no enunciado (i.e. Input Network Throughput, Output Network Throughput e Available Output Bandwidth), mas depois de recebermos os dados de exemplo, baseámo-nos nesses para especificar o problema da forma seguinte:

Variáveis	***Range***
Memory Usage	[0, 1]
Processor Load	[0, 1]
Input Network Throughput	[0, 1]
Output Network Throughput	[0, 1]
Available Output Bandwidth	[0, 1]
Latency	[0, 1]
CLP Variation	[-1, 1]

TABLE I: Especificar o problema

III. Termos linguísticos

Para todas as variáveis de input, acabámos por escolher os termos linguísticos *low*, *medium* e *high*.

Para a variável de *output*, testámos os dois casos:

- *increase* e *decrease*
- *increase*, *maintain* *decrease*

IV. Determinar as variáveis fuzzy

Para **todas** as nossas variáveis de ***input*** utilizámos funções de *membership* triangulares com o seguinte aspeto

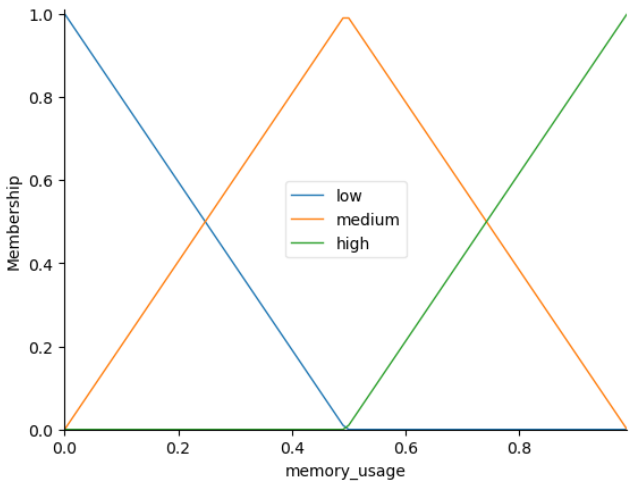

Fig. 1: Ilustração da função de *membership* utilizada para as nossas variáveis de input (neste caso, para a variável *Memory Usage*)

Já para a variável de ***output***, testámos os dois casos (*increase* e *decrease* vs *increase*, *maintain* *decrease*). Mais ainda,

acrescentámos um parâmetro que designámos disparidade que reflete o quanto ajustaríamos cada conjunto *fuzzy*. Depois optimizámos este parâmetro com os dados que recebemos para melhor definir esta tão importante função:

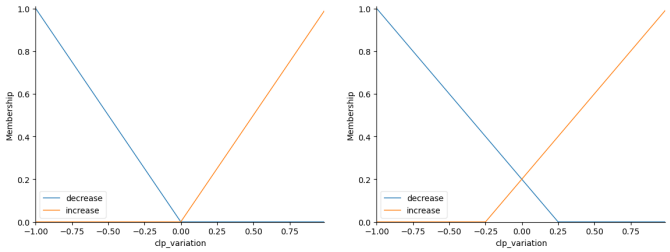

Fig. 2: Função de *membership* de *CLP Variation* de **2** termos linguísticos com disparidade = 0 (esquerda) e disparidade = 0.25 (direita)

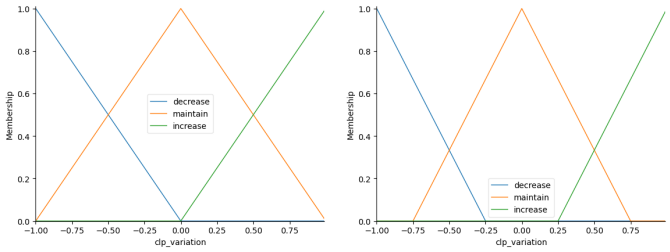

Fig. 3: Função de *membership* de *CLP Variation* de **3** termos linguísticos com disparidade = 0 (esquerda) e disparidade = 0.25 (direita)

Nota: O método de *defuzzification* é o centróide.

V. Importância das variáveis

Para prevenir o problema de explosão combinatória, selecionámos de início as variáveis que considerámos relevantes para o problema (por ordem de relevância):

1) *Processor Load*: Este é geralmente o parâmetro mais crítico no que toca a determinar o desempenho de um dispositivo. Quando tem carga a mais, causa problemas ao sistema.
2) *Memory Usage*: O uso de memória também importante, visto que afeta a capacidade dos dispositivos conseguirem lidar com tráfego de entrada e saída. Se esta estiver quase cheia, atrasará o processamento dos dados.
3) *Latency*: A latência é um parâmetro importante relativamente ao desempenho da rede, pelo que geralmente é um sintoma de outros problemas (por exemplo, relativo às duas variáveis acima). Pode ser uma boa forma de medir ”empates”.
4) *Available Output Bandwidth*: A largura de banda de saída disponível tem um papel significativo no desempenho do nosso sistema, embora não seja o parâmetro mais crítico. Esta influencia diretamente a experiência do usuário e o desempenho geral do sistema. Por exemplo, se a largura de banda de saída for baixa, o sistema pode ficar congestionado com dados à espera de serem enviados, o que pode atrasar o sistema e afetar a sua capacidade de processar novos pedidos.

As restantes variáveis de *Input* e *Output Network Throughput* não são tão relevantes para o nosso sistema. Estas representam a quantidade de dados que estão a ser enviados e recebidos através da rede. Embora possam afetar o desempenho de um sistema informático e a qualidade da ligação à Internet, o seu impacto pode não ser tão direto ou tão significativo como o impacto das outras variáveis. Por exemplo, um sistema pode ter um débito de rede elevado mas, mesmo assim, ter um desempenho fraco se a carga do processador for elevada ou a utilização da memória estiver próxima da capacidade.

VI. Construção das regras fuzzy

Como nota inicial, acabámos por criar poucos conjuntos que utilizam os **3** termos linguísticos de *CLPV* (apenas os das tabelas II, III e V). Decidimos de modo geral só usar **2** para os restantes porque facilitava a criação de regras corretas (sobretudo tendo em conta que não somos peritos na área e os dados que recebemos não deixam de ser poucos).

A. Regras iniciais

O primeiro conjunto de regras que criámos foi baseado nas 4 variáveis que consideramos mais importantes:

- *MU* alta $\wedge$ *PL* alta $\Rightarrow$ *CLPV* diminui
- *MU* baixa $\wedge$ *PL* baixa $\Rightarrow$ *CLPV* aumenta
- *Latency* baixa $\vee$ *Bandwidth* baixa $\Rightarrow$ *CLPV* diminui
- *Latency* alta $\vee$ *Bandwidth* alta $\Rightarrow$ *CLPV* aumenta

Este foi o que nos pareceu mais intuitivo ao início. Como utilizámos o pacote *skfuzzy* para implementar o sistema, este aplica interpolação para definir os restantes eventos possíveis, pelo que não passámos mais regras.

B. Regras a uma variável

Para este caso, apenas experimentámos usar as duas variáveis que consideramos mais importantes: *Processor Load* e *Memory Usage*:

	Memory Usage		
Terms	**Low**	**Medium**	**High**
Output: CLP Variation	Increase	Maintain	Decrease

TABLE II: Regras para o sistema com a variável *Memory Usage* (*CLP Variation* com **3** termos)

	Processor Load		
Terms	**Low**	**Medium**	**High**
Output: CLP Variation	Increase	Maintain	Decrease

TABLE III: Regras para o sistema com a variável *Processor Load* (*CLP Variation* com **3** termos)

C. Regras a duas variáveis

Em relação às regras de duas variáveis fizemos o que considerámos ter mais sentido, usar as duas variáveis mais importantes para o sistema.

No primeiro caso, para **2** termos linguísticos da variável de *output*:

Output: CLP Variation		Memory Usage		
		Low	Medium	High
Processor Load	Low	Increase	Increase	Decrease
	Medium	Increase	Increase	Decrease
	High	Decrease	Decrease	Decrease

TABLE IV: Regras para o sistema com duas variáveis *Memory Usage* e *Processor Load* (*CLP Variation* com **2** termos)

e no segundo caso, para **3** termos linguísticos da variável de *output*.

Output: CLP Variation		Memory Usage		
		Low	Medium	High
Processor Load	Low	Increase	Increase	Decrease
	Medium	Increase	Maintain	Decrease
	High	Decrease	Decrease	Decrease

TABLE V: Regras para o sistema com duas variáveis *Memory Usage* e *Processor Load* (*CLP Variation* com **3** termos), i.e. a única diferença é a sugestão do meio.

D. Regras a três variáveis

Para 3 variáveis, utilizámos a terceira variável mais importante (*Bandwidth*) como critério de desempate caso *Processor Load* e *Memory Usage* fossem médias:

- $(MU$ média $\wedge$ PL média$)$ $\wedge$ *Bandwidth* baixa $\Rightarrow$ *CLPV* diminui
- $(MU$ média $\wedge$ PL média$)$ $\wedge$ $(Bandwidth$ alta $\vee$ *Bandwidth* média$)$ $\Rightarrow$ *CLPV* aumenta

E. Combinações de sistemas: *método de Combs*

Finalmente, tentámos criar um método de ***ensemble*** que conseguisse incorporar as 4 variáveis que achamos mais importantes de forma interessante sem trazer problemas acrescidos devido a possivel demasiada complexidade. Assim sendo, criámos duas novas variáveis: ***Computacional Effort*** (*CompEff*) e ***Network Effort*** (*NetEff*), que achámos que poderiam traduzir bem o que se estava a passar no sistema *IoT*. Para *ComEff*:

- MU alta $\vee$ PL alta $\Rightarrow$ *CompEff* alto
- MU média $\wedge$ PL média $\Rightarrow$ *CompEff* médio
- MU baixa $\wedge$ PL baixa $\Rightarrow$ *CompEff* baixo
- $\Big((MU$ baixa $\wedge$ PL média$)$ $\wedge$ $(PL$ baixa $\wedge$ MU média$)\Big)$ $\Rightarrow$ *CompEff* baixo

Para *NetEff*:

- *Latency* alta $\vee$ *Bandwidth* baixa $\Rightarrow$ *CompEff* alto
- *Latency* média $\wedge$ *Bandwidth* média $\Rightarrow$ *CompEff* médio
- *Latency* baixa $\wedge$ *Bandwidth* alta $\Rightarrow$ *CompEff* baixo
- $\Big((Latency$ baixa $\wedge$ *Bandwidth* média$)$ $\wedge$ $(Bandwidth$ alta $\wedge$ *Latency* média$)\Big)$ $\Rightarrow$ *CompEff* baixo

Output: CLP Variation		NetEff		
		Low	Medium	High
Processor Load	Low	Increase	Increase	Decrease
	Medium	Increase	Increase	Decrease
	High	Decrease	Decrease	Decrease

TABLE VI: Regras para o sistema baseado no método de *Combs* (*CLPV* tem **2** termos linguísticos)

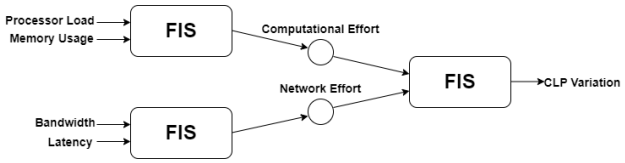

Fig. 4: Ilustração do nosso sistema baseado no método de *Combs*

VII. Criar o sistema de inferência

Como mencionado antes, para criar o nosso sistema de inferência *fuzzy*, recorremos ao pacote *skfuzzy*. Aqui seguem dois exemplos de output obtidos para as regras da tabela IV (figuras 5 e 6):

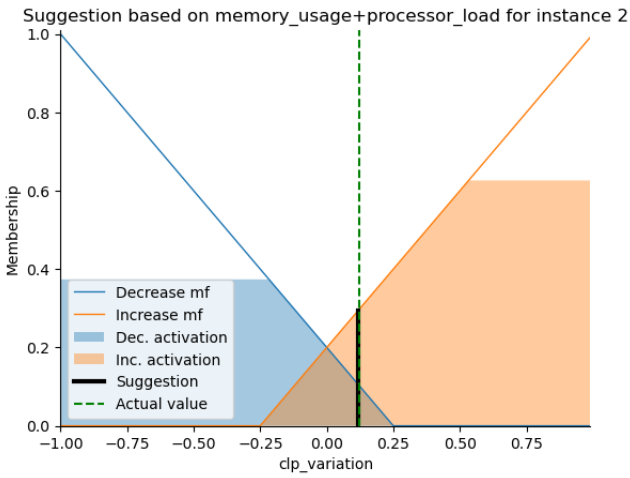

Fig. 5: *Output* para um instante, pelas regras da tabela IV

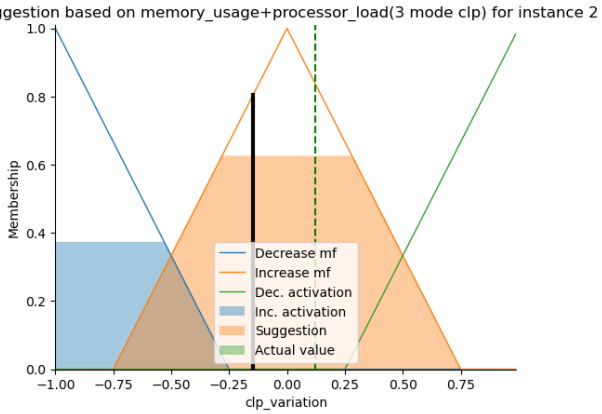

Fig. 6: *Output* para um instante, pelas regras da tabela V

A linha a preto representa a sugestão do modelo *fuzzy*, enquanto a linha a tracejado verde representa o verdadeiro valor de *CLPV*.

VIII. Avaliação de resultados e afinar o sistema

Para avaliar a performance de cada regra, tínhamos primeiramente feito uma tabela rápida para um valor fixo de disparidade a todas as regras (pode ser vista na **última página**, a tabela VII.

No entanto, queríamos algo melhor, sobretudo tendo em conta os poucos dados obtidos. Assim sendo, utilizámos as medidas *RMSE* e *MSE* nas sugestões de *CLPV* previstas.

O *Mean Squared Error* (MSE) é das métricas mais comummente utilizadas para analisar o desempenho de modelos de regressão (que é o que o nosso *FIS* acaba por ser). Esta métrica é calculada da seguinte forma:

$$MSE = \frac{1}{N}\sum_{i=1}^{N}(y_i - \hat{y}_i)^2 \tag{1}$$

em que y_i representa os valores reais, $\hat{y}_i$ representa os valores previstos, e N é o número total de observações.

O *Root Mean Square Error* (RMSE) é derivado do *MSE*, aplicando-lhe a raiz quadrada:

$$RMSE = \sqrt{MSE} \tag{2}$$

A utilização do *RMSE* e do *MSE* para um problema de regressão com poucos dados é benéfica porque fornece uma avaliação abrangente do desempenho do modelo. Enquanto o *MSE* enfatiza os erros maiores e ajuda a identificar os outliers, o RMSE oferece uma medida facilmente interpretável nas unidades originais dos dados.

Ao considerar ambas as métricas, podemos obter uma melhor compreensão da magnitude média dos erros e identificar desvios extremos. Esta abordagem combinada permite uma avaliação com mais nuance da exatidão do modelo e ajuda a descobrir os seus pontos fortes e fracos no cenário de dados limitados.

Portanto, depois fizemos a disparidade variar de -0.9 a 0.9 (0.01 a 0.01) e avaliamos esta média. Assim sendo, a métrica de performance que realmente considerámos foi a média de ambos,que é a medida que podemos observar na figura 7 (também **última página**).

IX. Conclusão

Neste projeto, desenvolvemos com sucesso um Sistema de Inferência *Fuzzy* (FIS) para prever a variação do *CLP* (Parâmetro de Carga Computacional) com base em diversas variáveis de entrada. As variáveis consideradas foram o uso de memória e a carga do processador, que foram identificados como os fatores mais significativos que influenciam a variação do CLP.

O FIS foi concebido utilizando um conjunto de regras derivadas do conhecimento especializado e da intuição sobre o sistema. Estas regras foram depois utilizadas para criar funções de associação, que mapeiam as variáveis de entrada para um conjunto difuso que representa a variável de saída, a variação do CLP.

O sistema foi testado utilizando um pequeno conjunto de dados de 10 instâncias. Apesar do pequeno tamanho do conjunto de dados, o FIS foi capaz de fornecer previsões razoáveis para a variação do CLP. Isto demonstra o potencial da lógica *fuzzy* para modelar sistemas complexos com dados limitados.

No entanto, é importante notar que o desempenho do FIS poderia ser melhorado com mais dados. O pequeno tamanho do conjunto de dados limitou a capacidade de validar completamente o modelo e ajustar os seus parâmetros.

Em conclusão, este projeto demonstrou que a lógica *fuzzy* é uma abordagem promissora para modelar a variação do *CLP* num sistema *IoT*. O FIS desenvolvido fornece uma base para trabalhos futuros nesta área, com potenciais aplicações na afetação de recursos, otimização de sistemas e manutenção preditiva.

TABLE VII: Tabela com primeiros resultados, utilizando disparidade = 0.25 para todas as regras (nota: as regras estão ordenadas tal como foram inicialmente apresentadas)

Rule	Variables used	CLP variation mode	Disparity	Performance
0	memory_usage, processor_load, bandwidth, latency	2	0.25	0.539870
1	processor_load	3	0.25	0.515668
2	memory_usage	3	0.25	0.593033
3	memory_usage, processor_load	2	0.25	0.371449
4	memory_usage, processor_load	3	0.25	0.544348
5	memory_usage, processor_load, output_bandwidth	3	0.25	0.340585
6	combs_method	2	0.25	0.817394

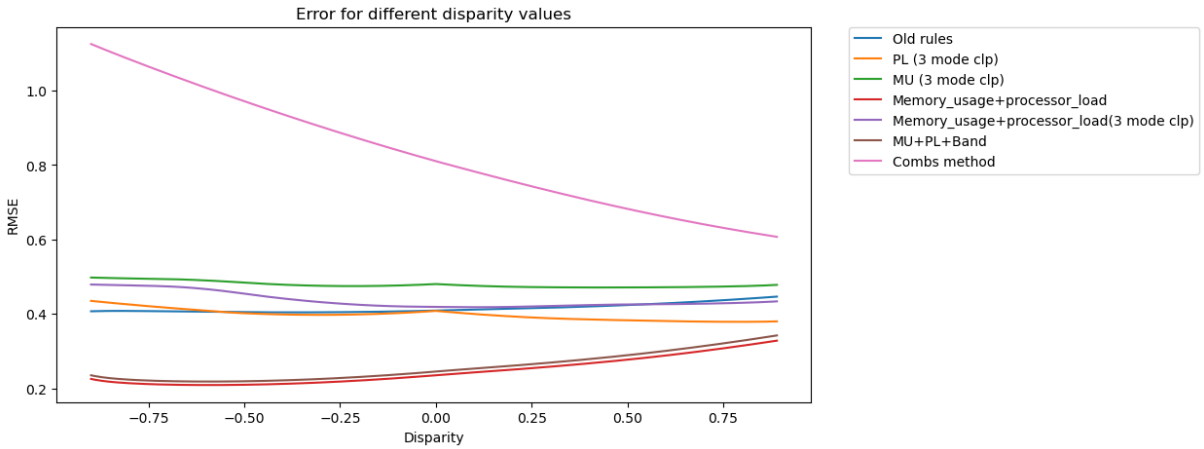

Fig. 7: *Performance* para diferentes valores de disparidade, para todas as regras utilizadas (**nota:** os *labels* estão ordenados de acordo com a forma com que apresentámos as regras ao longo do trabalho, ver na secção VI)