Trabajo Práctico: Fuzzy Logic

Juan Cruz De La Torre y Bautista Marelli

4 de Junio, 2020

1 Enunciado

Dada la situación actual de cursado virtual, muchas materias optan por realizar evaluaciones en forma de trabajos prácticos. Para asignar una nota a un trabajo, las cátedras suelen evaluar distintos aspectos del mismo y luego combinar esas evaluaciones (usualmente de una manera algo subjetiva) para generar una nota final.

Caso de estudio: decidimos concentrarnos en un formato de trabajo práctico algo usual en la carrera LCC. Se pide a los estudiantes realizar un trabajo que involucra el desarrollo de algún tipo de software. La evaluación del mismo se divide en dos instancias:

- en la primera, se tiene en cuenta la complejidad del trabajo, el rendimiento y la cantidad de errores encontrados.
- en la segunda, una vez obtenida una nota parcial del código entregado, se tiene una instancia oral con el alumno a modo de defensa del trabajo práctico a la cual se le asigna una nota. Contando con ambas notas, se decide una nota final del trabajo práctico.

La complejidad y el rendimiento son medidos como un valor entre 0 y 10, son descriptos como complejidad baja, media o alta y rendimiento pobre, medio o bueno de acuerdo con los conjuntos borrosos que se muestran a continuación.

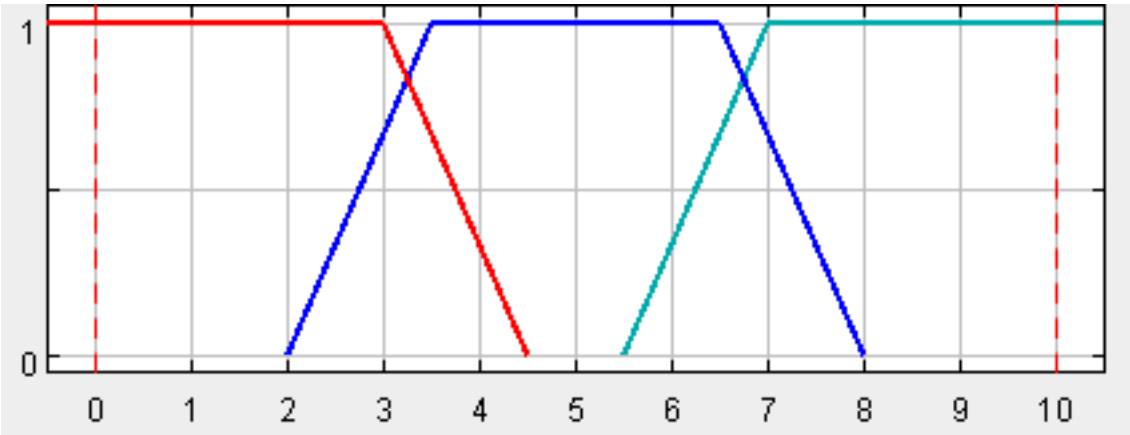

Figure 1: Conjuntos borrosos sobre complejidad (baja, media, alta)

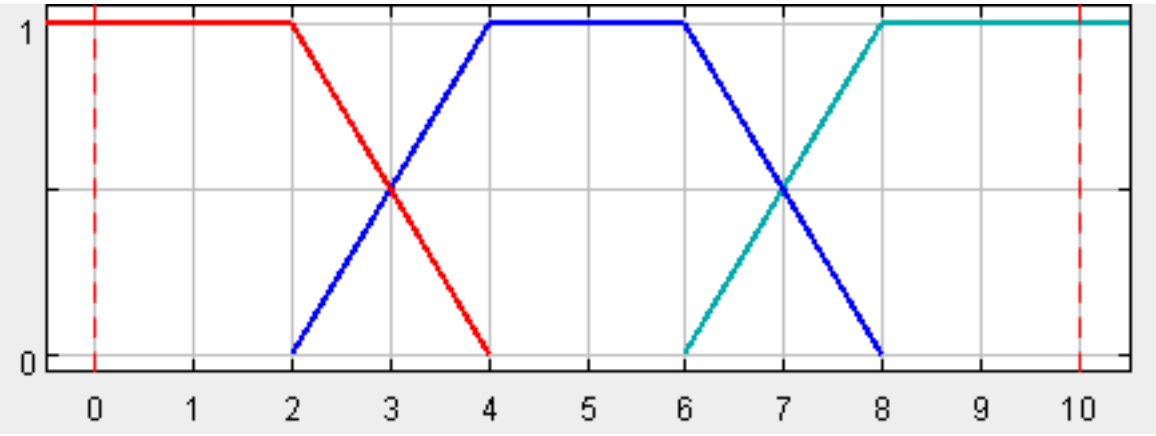

Figure 2: Conjuntos borrosos sobre rendimiento (pobre, medio, bueno)

Sobre la cantidad de errores, se puede decir si un trabajo tiene muy pocos, pocos, o muchos errores de acuerdo con los conjuntos borrosos:

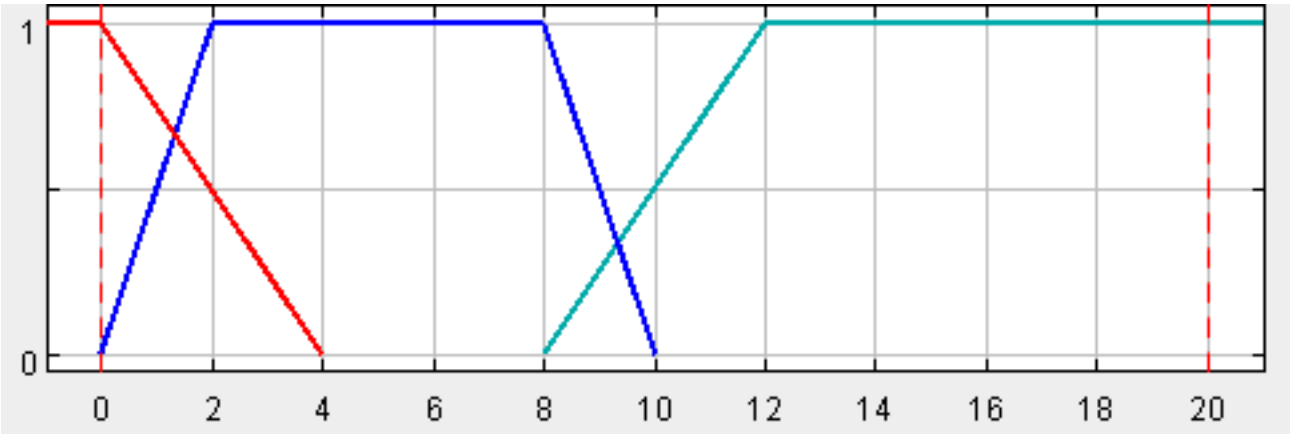

Figure 3: Conjuntos borrosos sobre errores (muy pocos, pocos, muchos)

La nota parcial del código es un valor entre 0 y 10 calificado, similarmente, como una nota Mala, Regular o Excelente de acuerdo a la figura 4 y las reglas que se utilizan se detallan a continuación:

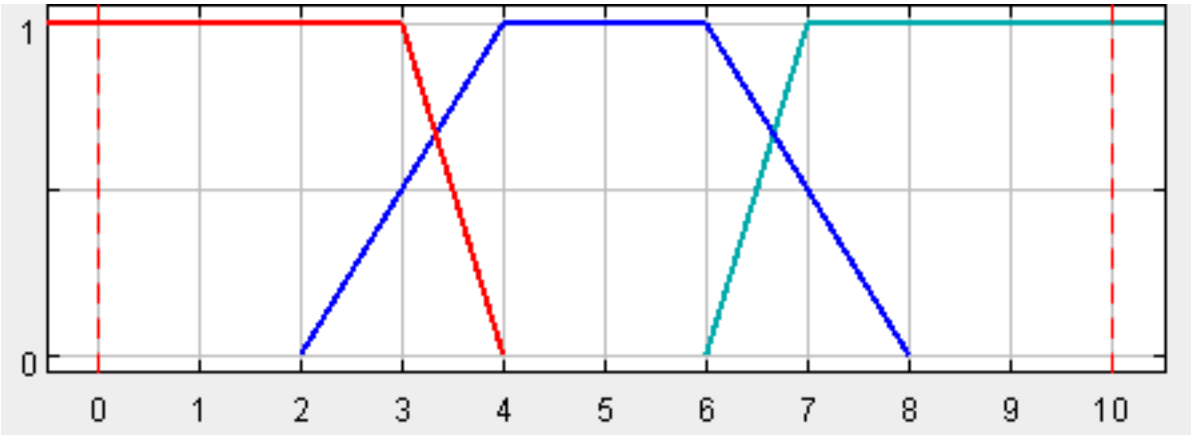

Figure 4: Conjuntos borrosos sobre la nota parcial del código (mala, regular, excelente)

- Si la complejidad es **baja** y hay **muchos** errores, la nota es **mala**.
- Si la complejidad es **baja** y hay **pocos** errores, entra en juego el rendimiento. Si el rendimiento es **pobre** la nota es **mala**, y si no la nota es **regular**.
- Si la complejidad es **baja** y hay **muy pocos** errores, la nota es **regular** excepto si el rendimiento es **bueno** en cuyo caso la nota es **excelente**.
- Si la complejidad es **media**, hay **muchos** errores y el rendimiento es **pobre**, la nota es **mala**. Si en cambio el rendimiento es **medio** o **bueno**, la nota es **regular**.
- Si la complejidad es **media** y hay **pocos** errores, la nota es **regular**.
- Si la complejidad es **media** y hay **muy pocos** errores, la nota es **excelente** excepto si el rendimiento es **pobre**, en cuyo casola nota es **regular**.
- Si la complejidad es **alta**, hay **muchos** errores y el rendimiento es **pobre**, la nota es **mala**. Si en cambio el renidmiento es **medio** o **bueno**, la nota es **regular**.
- Si la complejidad es **alta**, hay **pocos** errores y el rendimiento es **pobre**, la nota es **regular**. Si el rendimiento es **medio** o **bueno**, la nota es **excelente**.
- Y por último, si la complejidad es **alta** y hay **muy pocos** errores, la nota es **excelente**.

Para la segunda etapa, se tiene en cuenta tanto el valor obtenido de la nota parcial del código como una nota entre 0 y 10 que corresponde a la defensa, y que se califica también como mala, regular o excelente de acuerdo con los conjuntos borrosos:

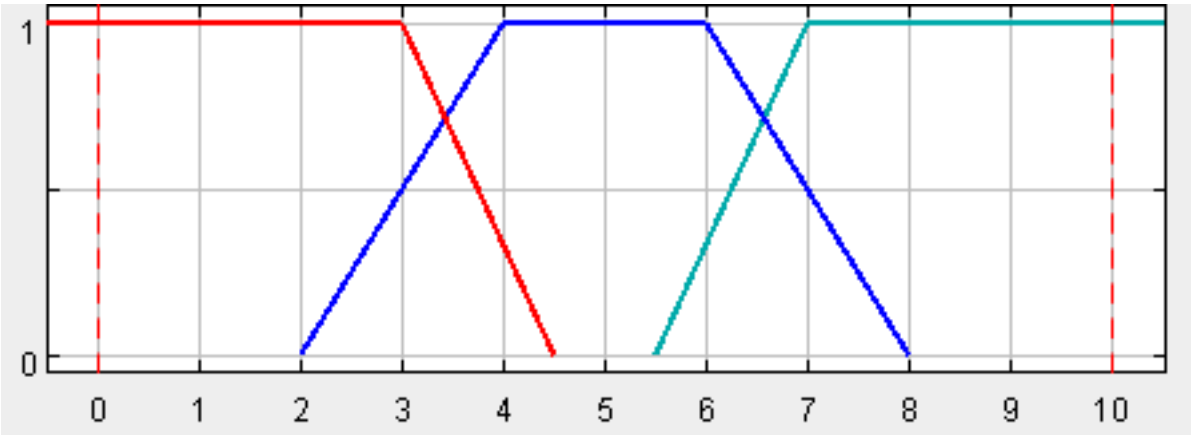

Figure 5: Conjuntos borrosos sobre la nota de la defensa (mala, regular, excelente)

Finalmente, la nota final del trabajo es un valor entre 0 y 10 también calificado como mala, regular o excelente de acuerdo a los conjuntos borrosos de la figura 6 y las reglas que se usan se detallan a continuación:

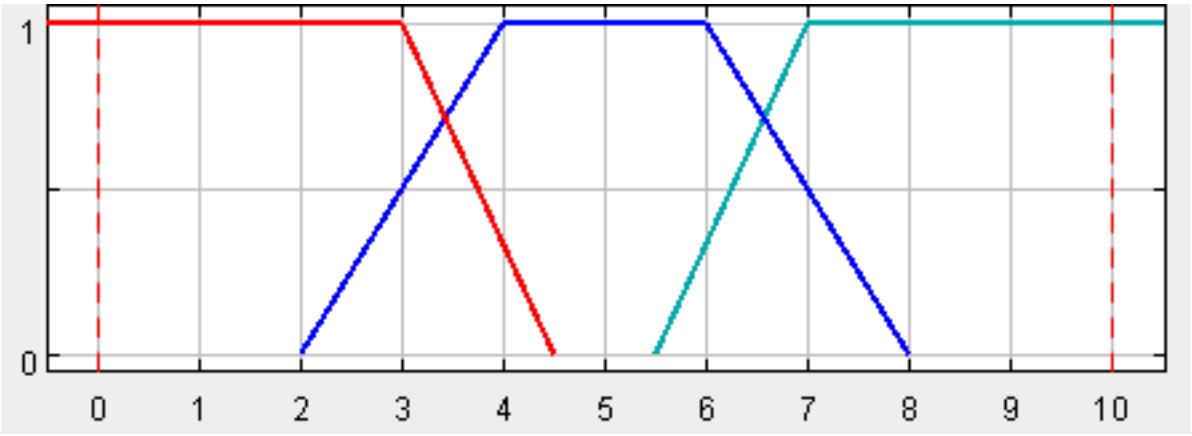

Figure 6: Conjuntos borrosos sobre la nota final (mala, regular, excelente)

- Si la nota parcial del código es **mala**, la nota final es **mala**.
- Si la nota parcial del código es **regular** y la defensa es **mala**, la nota final es **mala**.
- Si la nota parcial del código es **regular** y la defensa es **regular** o **excelente**, la nota final es **regular**.
- Si la nota parcial del código es **excelente**, y la defensa es **mala** o **regular**, la nota final es **regular**.
- Si la nota parcial del código es **excelente** y la defensa es **excelente**, entonces la nota final es **excelente**.

2 Ejemplos

Ejemplo 1

Para el primer ejemplo, consideramos un trabajo con una complejidad algo alta de 7.4, con 1 error y un rendimiento relativamente pobre de 2,6 puntos. En las figuras a continuación se muestran las reglas que se disparan, resultando en un output defuzzificado de 7,76. Esto es un resultado similar al que uno esperaria:

un trabajo de complejidad alta, con muy pocos errores tiene una nota excelente, a pesar del rendimiento pobre.

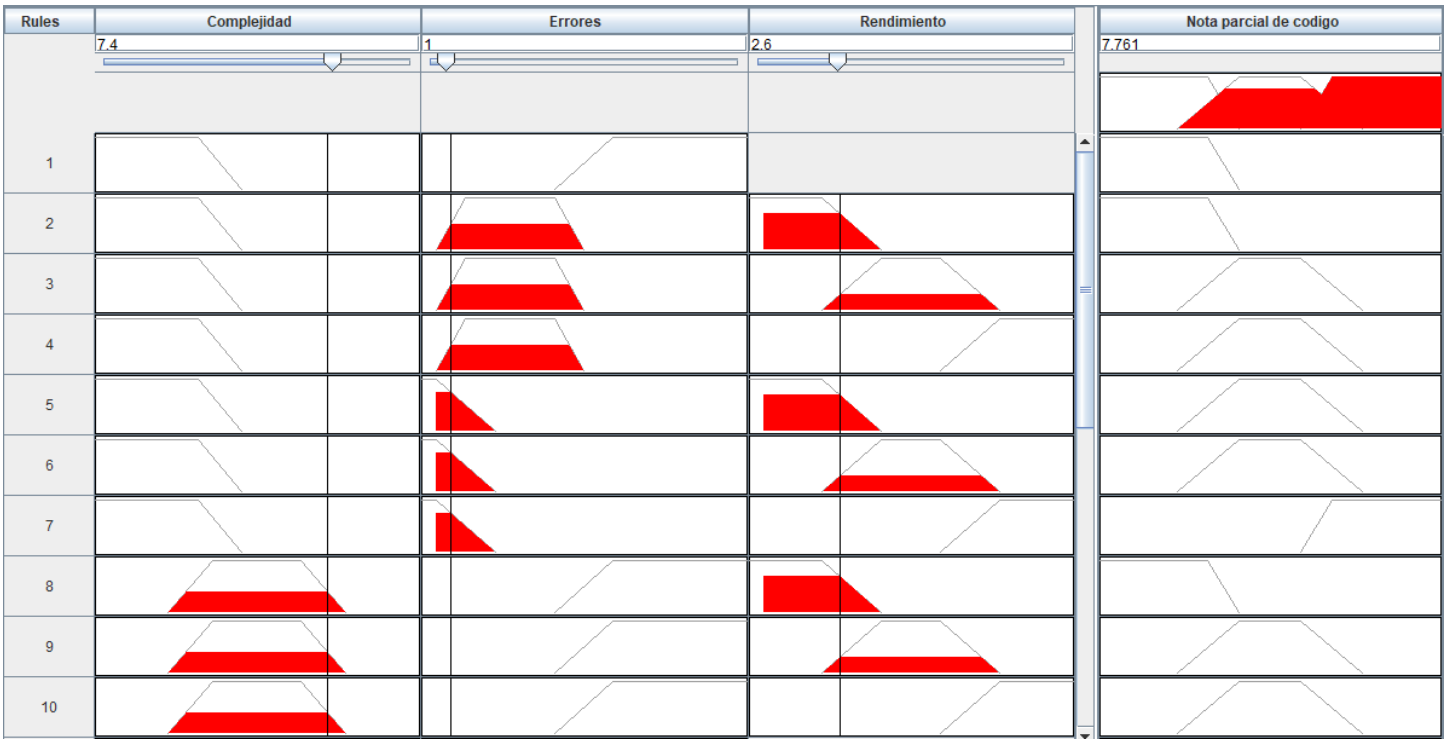

Figure 7: Ejemplo 1 - Etapa 1.a)

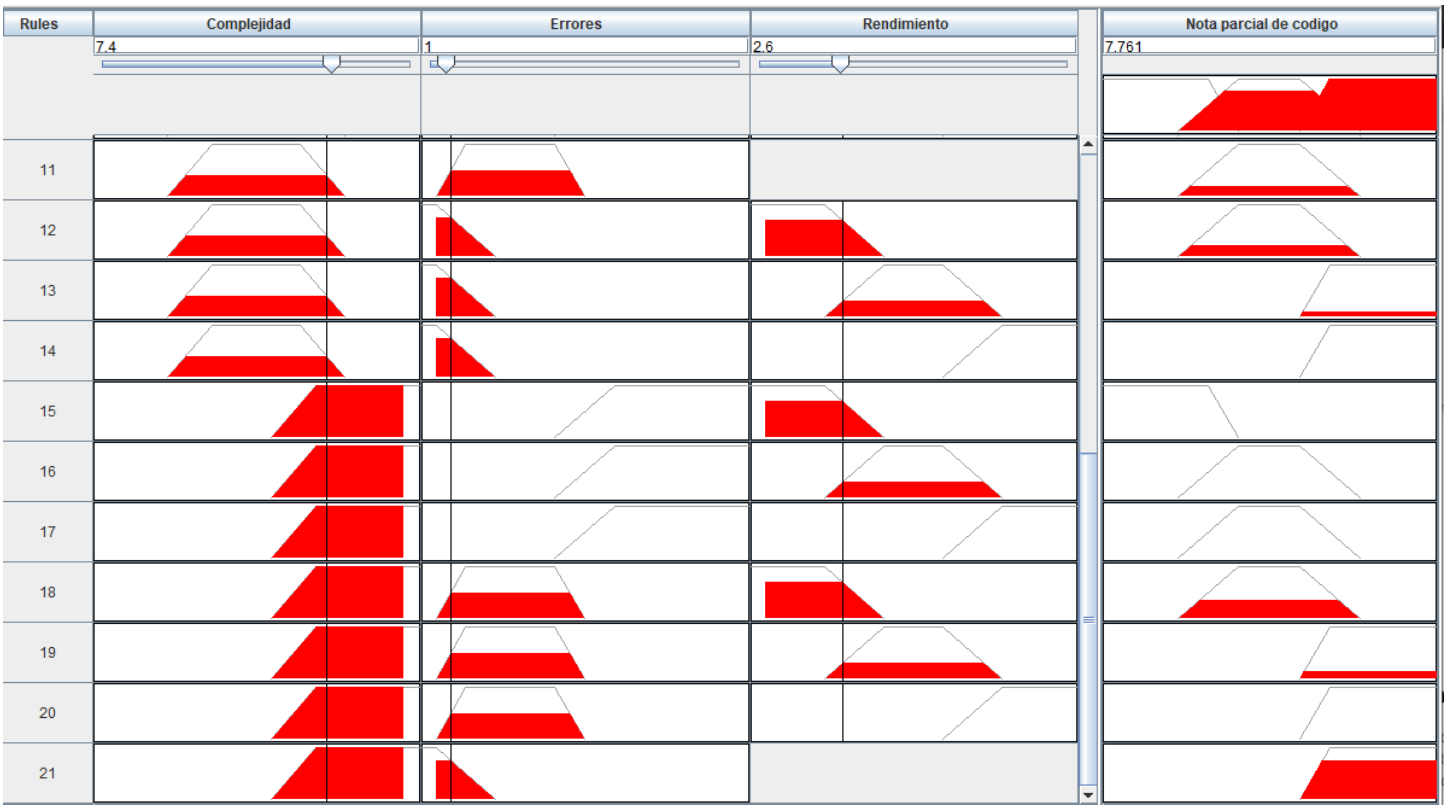

Figure 8: Ejemplo 1 - Etapa 1.b)

En la segunda etapa, se toma el valor obtenido como salida de la etapa anterior y se considera una

defensa entre regular y excelente de 6,5 puntos. A continuación se muestran las reglas que se disparan, y se obtiene una nota final de 7,3. Lo que uno esperaría es precisamente esto, ya que una nota parcial del código excelente y una defensa entre regular y excelente debería tener una nota final entre regular y excelente.

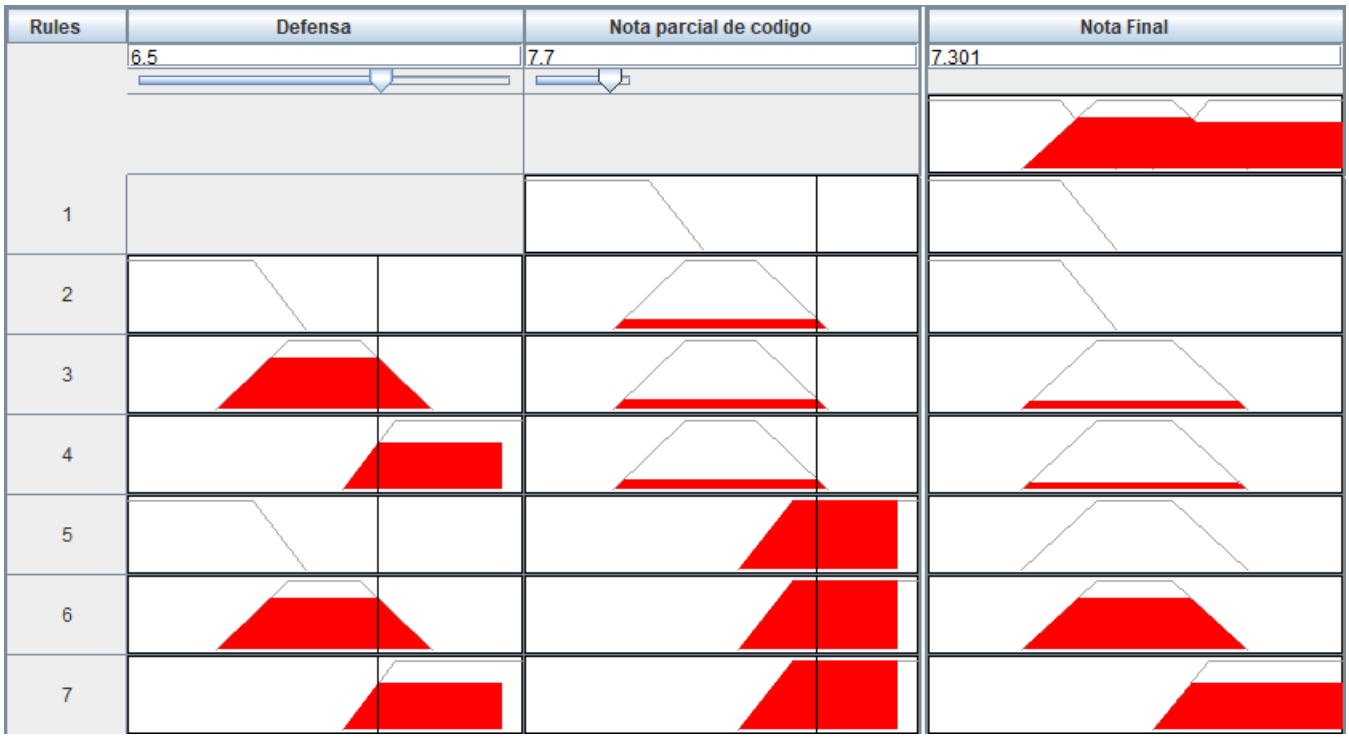

Figure 9: Ejemplo 1 - Etapa 2

Ejemplo 2

Para el segundo ejemplo, consideramos un trabajo al que se le asigna una complejidad de 3.2, con 2 errores y un muy buen rendimiento de 10 puntos. En las figuras a continuación se muestran las reglas que se disparan, resultando en un output defuzzificado de 7,4. Esto es un resultado similar al que uno esperaria: un trabajo de complejidad media, con pocos errores y muy buen rendimiento tiene una nota excelente. De acuerdo a las reglas, la nota debería ser regular; sin embargo, debido a el uso de la suma en lugar del máximo como T-conorma y el promedio como el método de defuzzificado la nota se ve desplazada levemente hacia el excelente (en la conclusión se habla un poco más sobre esto).

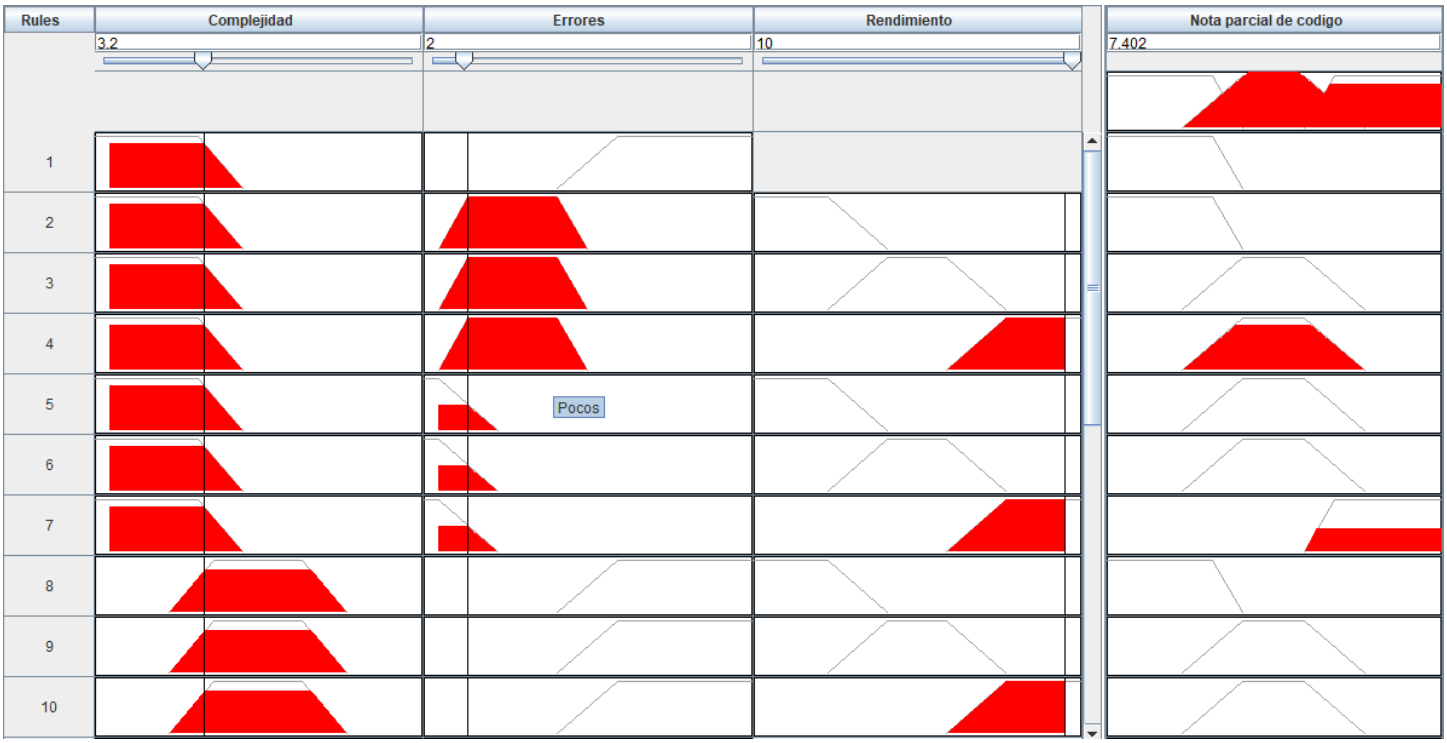

Figure 10: Ejemplo 2 - Etapa 1.a)

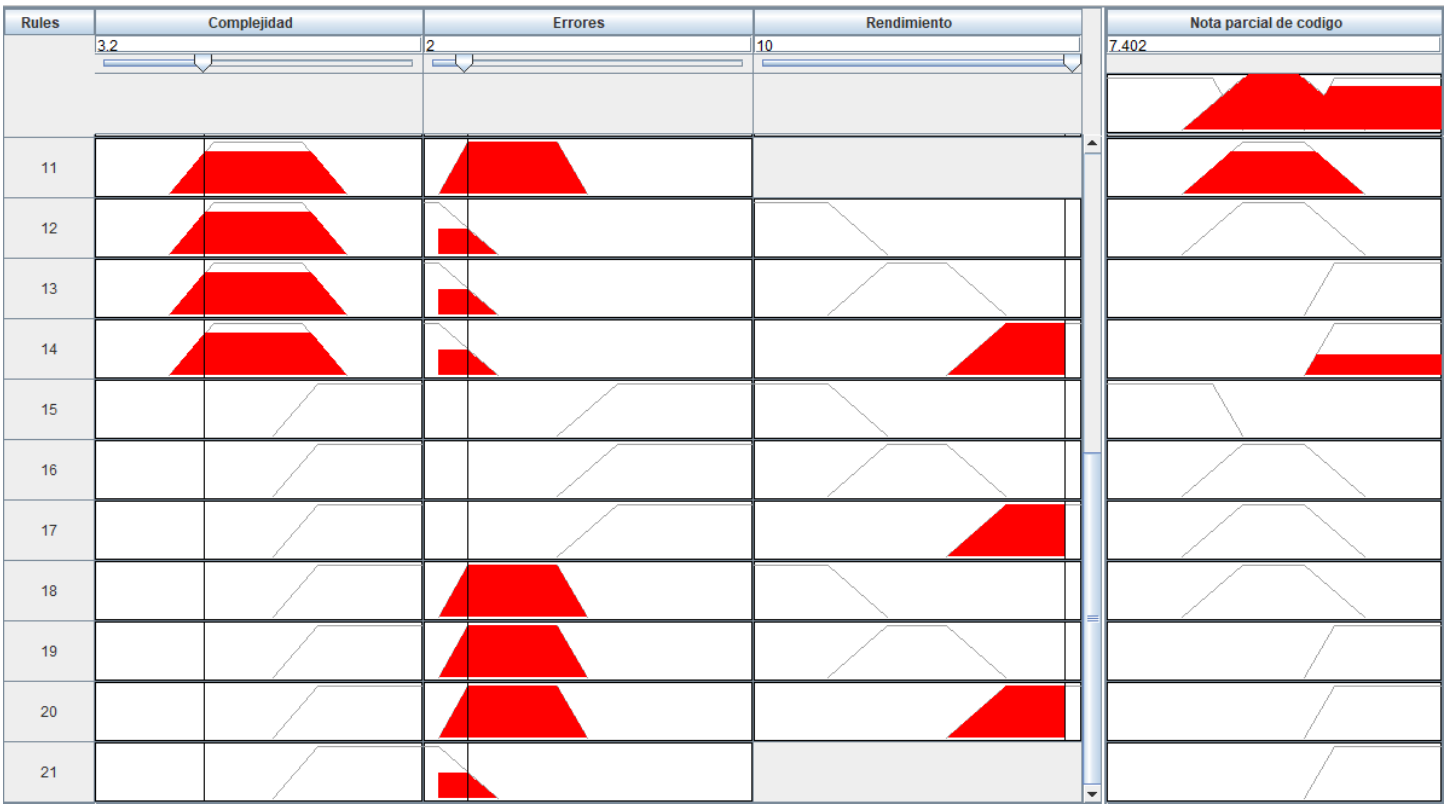

Figure 11: Ejemplo 2 - Etapa 1.b)

Para la segunda etapa, se toma el valor obtenido como salida de la etapa anterior y se considera una defensa entre mala y regular calificada como 3,5. Nuevamente se muestra en las figuras que siguen las reglas disparadas, y se obtiene un output de nota final similar al esperado de 3,87 (regular).

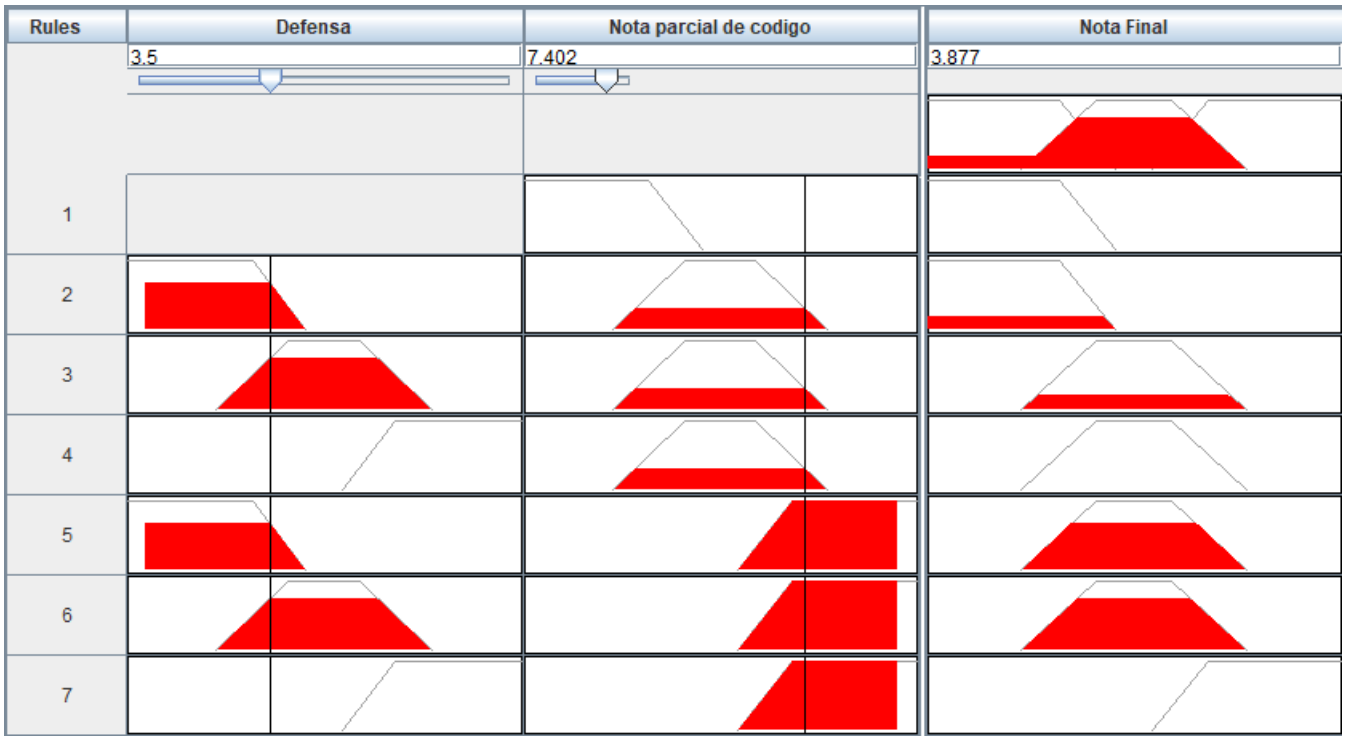

Figure 12: Ejemplo 2 - Etapa 2

3 Conclusión

A lo largo del trabajo práctico nos encontramos con varios obstáculos. Inicialmente planteamos una cantidad mayor de conjuntos borrosos para cada variable de entrada y salida y algunos criterios más, y si bien eso nos daba la posibilidad de ser más precisos con el output se volvía muy tedioso razonar sobre las reglas por su gran cantidad. Sumado a eso, más de una vez nos encontramos con que nuestras reglas no permitían inferir con algunos datos de entrada en particulares ya que el conjunto borroso de salida era no conexo; para solucionarlo, debimos prestar más atención a las reglas y cambiar las funciones de pertenencia de algunos de los conjuntos borrosos. Además, no ayudó que a veces FisPro decidía no actualizar correctamente los conjuntos borrosos e infería cosas incorrectas hasta reiniciarlo.

Otra situación a la que debimos enfrentarnos es parecida a la situación encontrada en la etapa 1 del segundo ejemplo: ¿son razonables los resultados inferidos dado un caso particular? Debimos experimentar con las reglas, los conjuntos borrosos y las T-normas y T-conormas y los métodos de defuzzificado para ver que combinación de ellas producía resultados más cercanos a lo esperado para diversos casos de prueba. Encontramos que, en general, usar el producto para la conjunción, la suma para la disjunción, y "área" como método de defuzzificado nos daba los mejores resultados a pesar de tender a desplazar hacia mal o excelente las notas que podrían estar dentro del regular.

Creemos que con preparación adecuada, mayor cantidad de conjuntos borrosos y variables de entradas como criterios, una cátedra podría dar uso a lógica borrosa para calcular notas de trabajos prácticos y otras instancias evaluativas. Utilizando reglas bien preparadas y conjuntos borrosos bien diseñados, se podría generar un sistema que infiera notas con una precisión mucho mayor a la que nosotros planteamos.