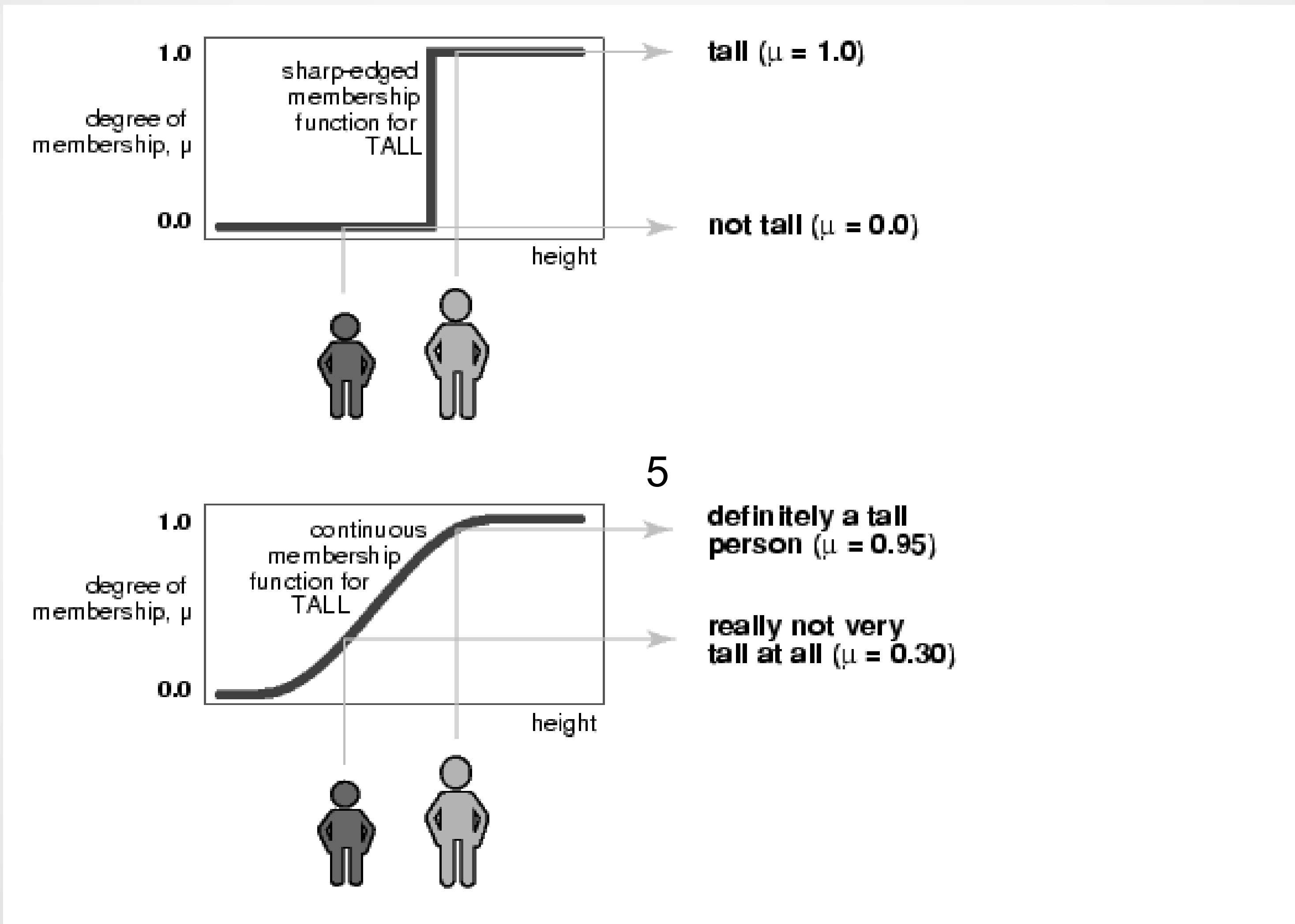
Logica fuzzy: fondamenti e alcune applicazioni

Mattia Antonino Di Gangi

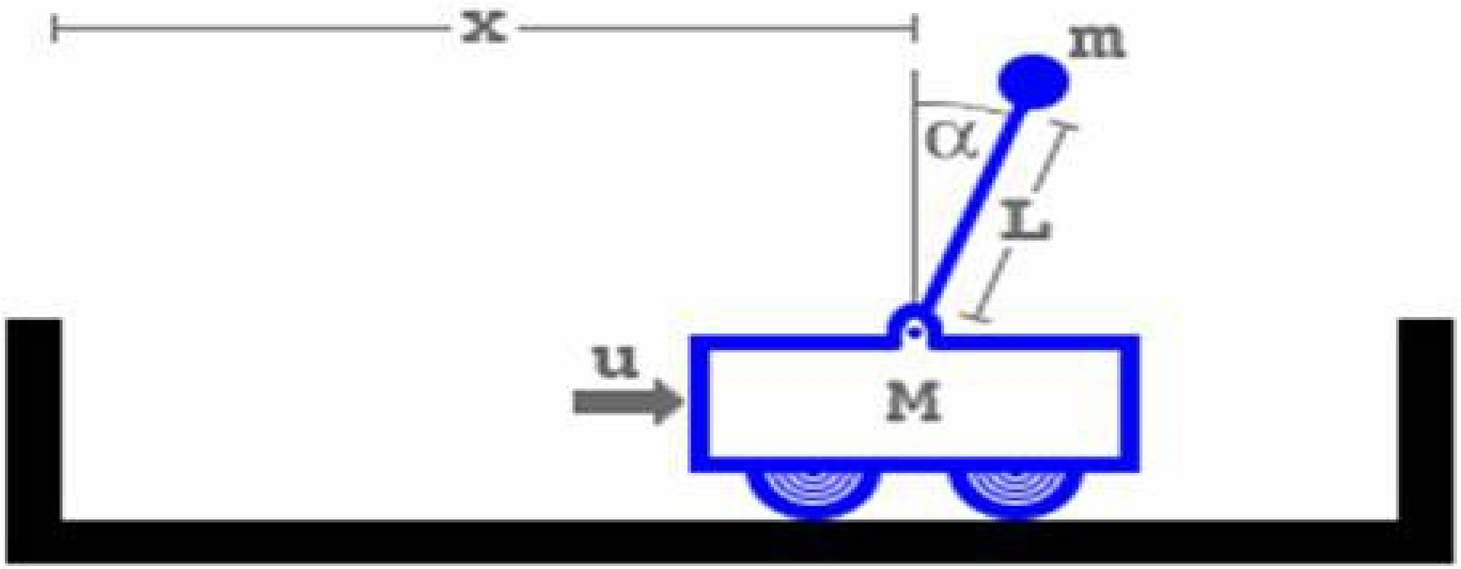
12 maggio 2015

mattiadigangi@gmail.com

Oltre la logica booleana



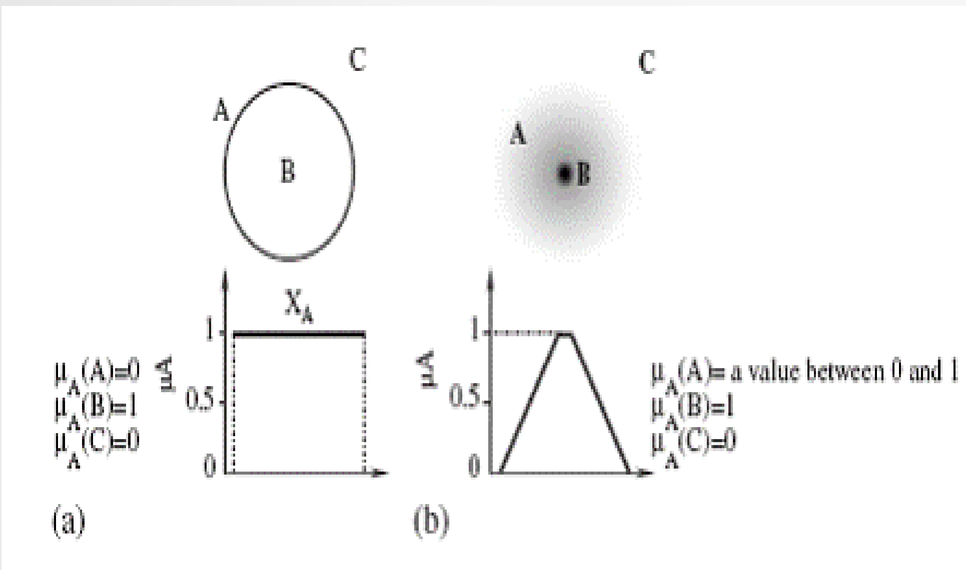
Rimodellare i problemi



$$\ddot{x} = \frac{u + mL\dot{\alpha}^2 \sin\alpha - mg\cos\alpha\sin\alpha}{M + m - m\cos^2\alpha}$$

$$\ddot{\alpha} = \frac{u\cos\alpha - (M+m)g\sin\alpha + mL\dot{\alpha}\cos\alpha\sin\alpha}{mL\cos^2\alpha - (M+m)L}$$

Insiemi Fuzzy

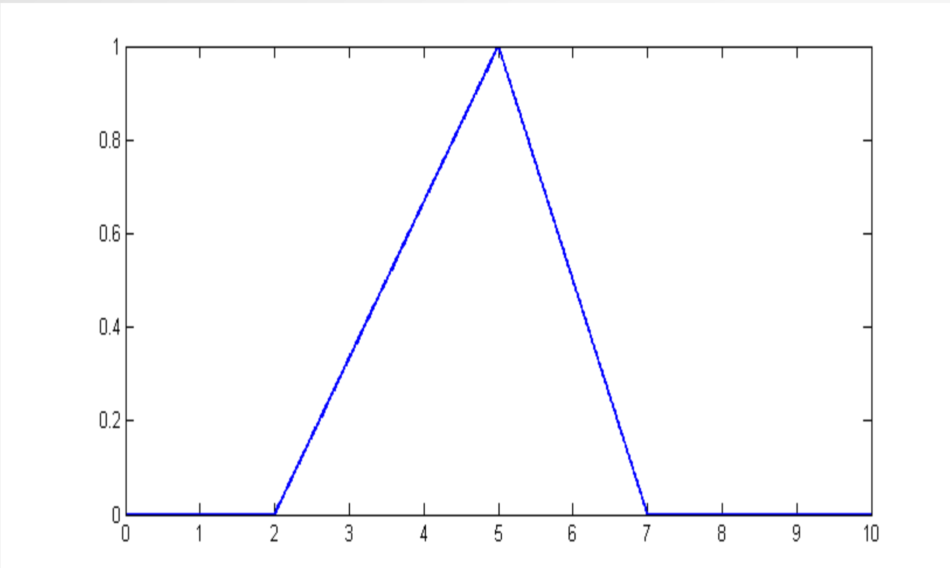


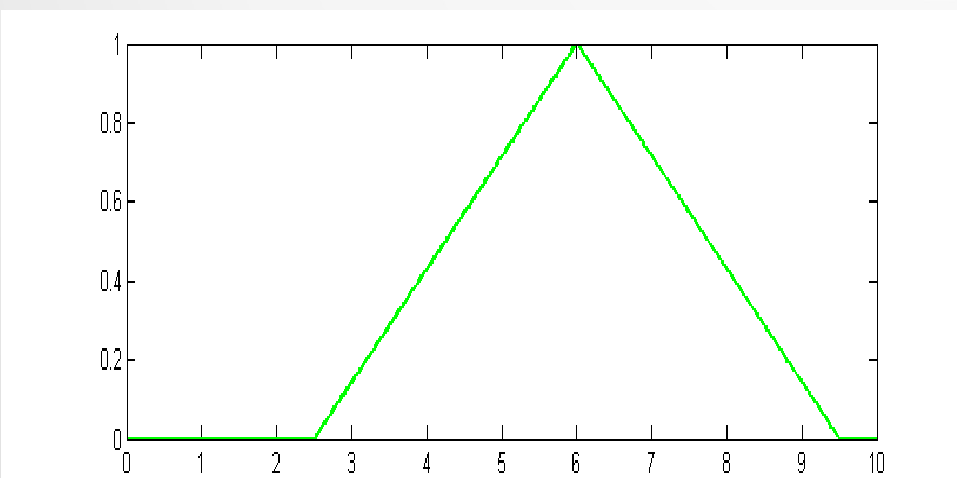
- Usati per rappresentare classi dai contorni non ben definiti.
- Definiti a partire da una funzione d'appartenenza: $\mu_A(x) : U \rightarrow L$
- E' importante il *significato* attribuito all'insieme.

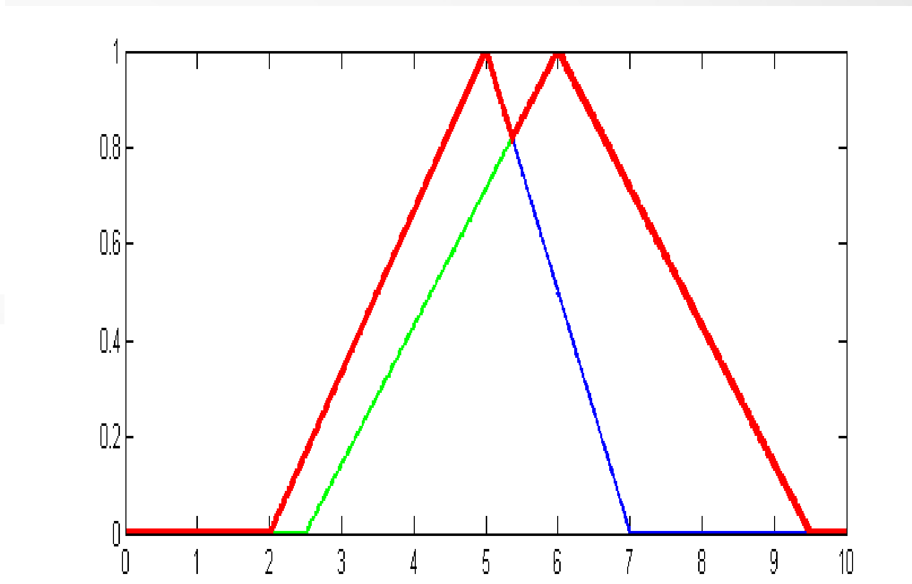
Operazioni tra insiemi

Come nella teoria classica degli insiemi, anche nella FST vogliamo poter effettuare unione, intersezione e complemento di insiemi. L'introduzione di valori reali apre la scena a diverse funzioni che possono adempiere a questi ruoli.

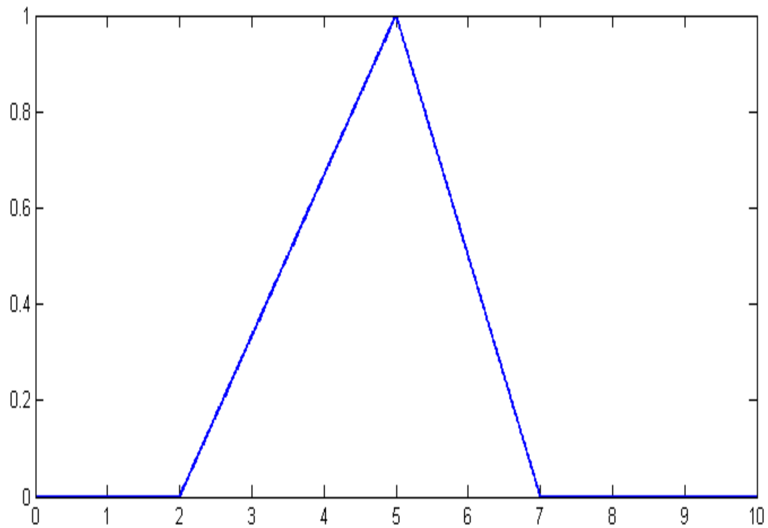
Unione

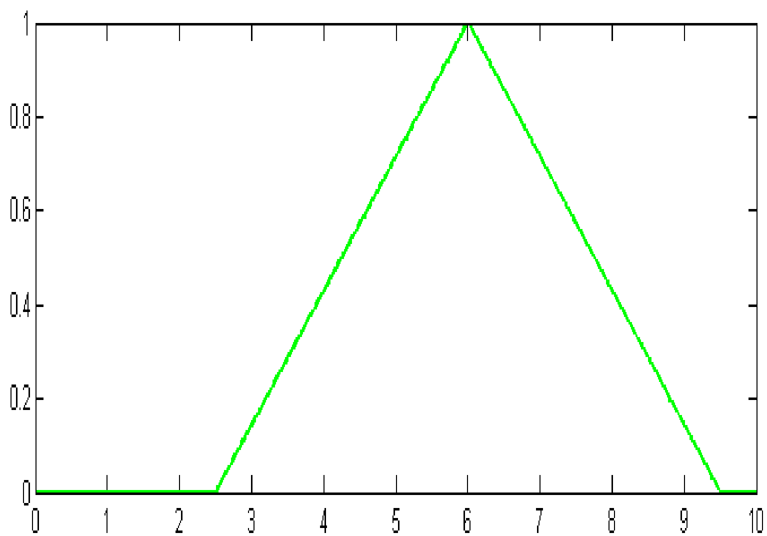

0 0.2 0.4 0.6 0.8 1
0 1 2 3 4 5 6 7 8 9 10

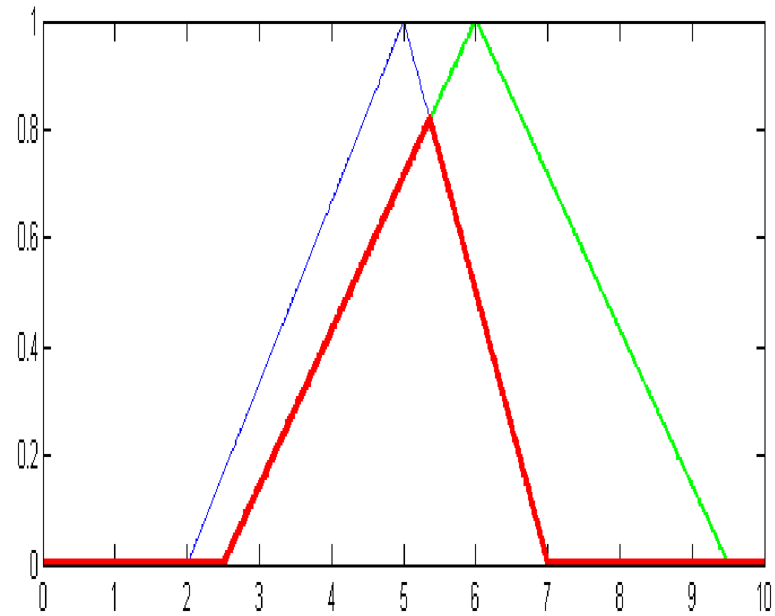

0 0.2 0.4 0.6 0.8 1
0 1 2 3 4 5 6 7 8 9 10


0 0.2 0.4 0.6 0.8 1
0 1 2 3 4 5 6 7 8 9 10

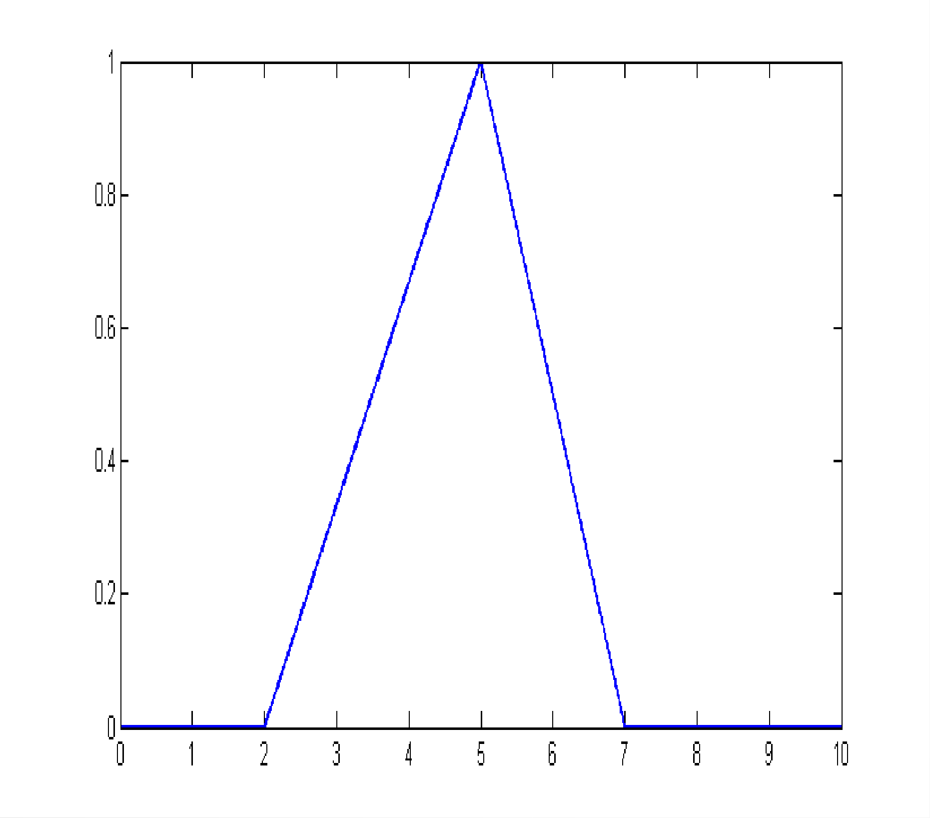
Intersezione

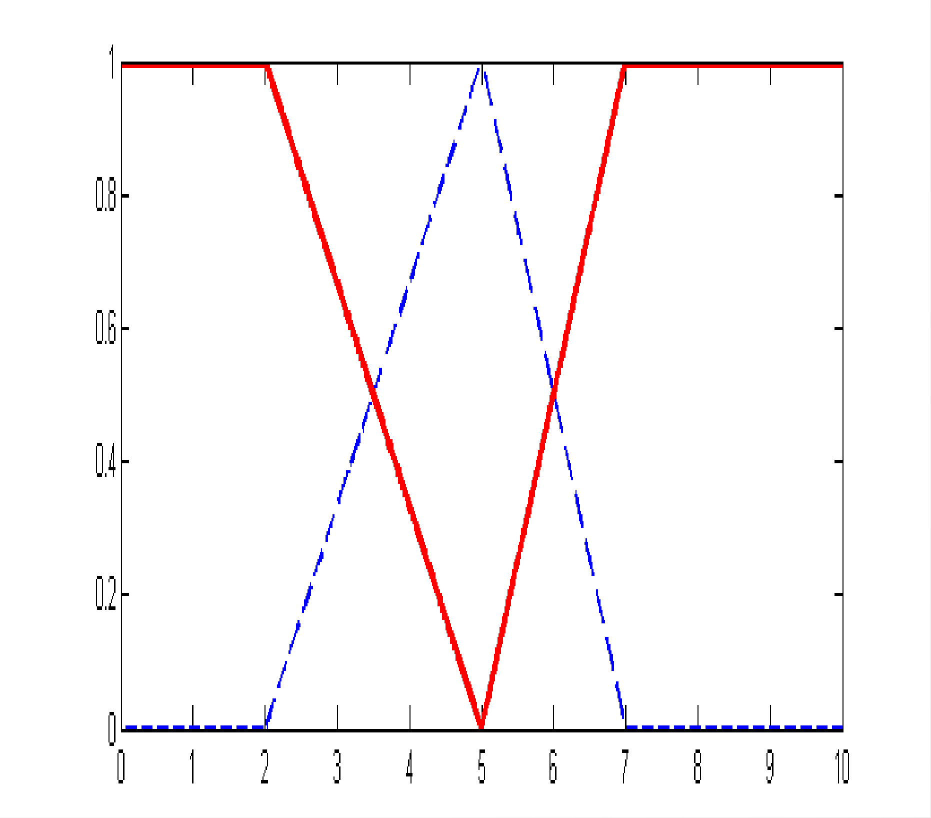






Complemento





Logica Fuzzy

- E' un sistema di calcolo e ragionamento che usa come oggetti insiemi fuzzy.
- Si basa sull'utilizzo di regole IF-THEN
- Permette di effettuare calcoli con valori approssimati.

```
IF PRESSURE IS HIGH AND
TEMPERATURE IS LOW
THEN PRECIPITATION IS RAIN
```

Congiunzione - T-norme

- **Intuizione:** Una congiunzione ha valore di verità alto quando entrambi i congiunti hanno un grado alto.

X	Y	X AND Y
0	0	0
0	1	0
1	0	0
1	1	0

- T-norma: Funzione F per cui vale:

  1) $F(x, 1) = x$

  2) $F(x, y) = F(y, x)$

  3) $F(x,F(y, z)) = F(F(x, y),z)$

  4) Se $x \leq u$ e $y \leq v$:

  $F(x,y) \leq F(u,v)$

Disgiunzione – T-conorme

- **Intuizione:** Il valore di verità è elevato quando almeno una delle proposizioni ha un grado elevato.

X	Y	X OR Y
0	0	0
0	1	1
1	0	1
1	1	1

- T-conorma: Una funzione F per cui vale:

1) $F(x, 0) = x$

2) $F(x, y) = F(y, x)$

3) $F(x,F(y, z)) = F(F(x, y),z)$

4) Se $x \leq u$ e $y \leq v$:

$F(x,y) \leq F(u,v)$

Negazione

- Funzione N tale che:

  1) $N(0) = 1$

  2) $N(1) = 0$

  3) N è non-crescente

X	Y
0	1
1	0

- Esistono numerosissime funzioni di negazione, di forme anche molto diverse. Le più “intuitive” sono le negazioni forti, cioè funzioni di negazione continue, strettamente decrescenti e involutive:

$$N(N(X)) = X$$

Implicazione

- Esistono diverse classi di funzioni che rispondono a questi criteri.

1: se $x \leq y$, allora $I(x; z) \geq I(y; z)$, cioè I è decrescente nella prima variabile;

2: se $y \leq z$, allora $I(x; y) \leq I(x; z)$, cioè I è crescente nella seconda variabile;

3: $I(1; 0) = 0$; $I(0; 0) = I(1; 1) = 1$

- **S-Implicazione:**

$$I(x; y) = S(N(x); y)$$

Es: $\max\{ 1-x, y\}$

- **R-Implicazione:**

$$I_T(x; y) = \sup \{z \mid T(x; z) \leq y\}$$

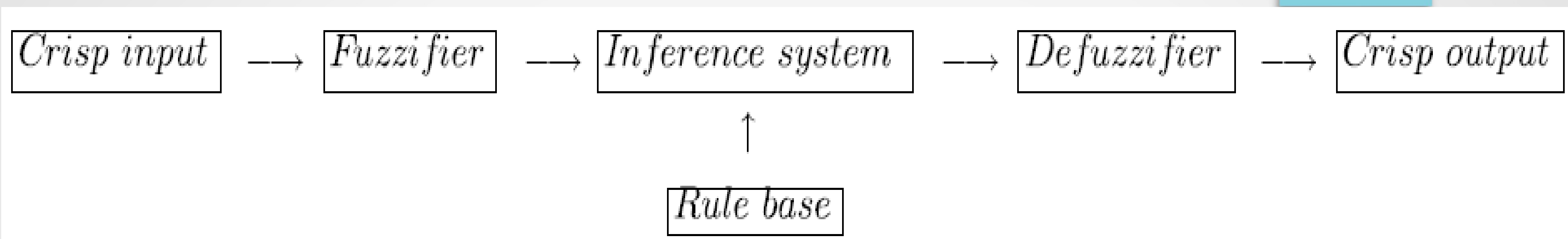
Es: $\sup \{z \mid \min\{x; z\} \leq y\}$

Condizionali

- Nonostante questa varietà di implicazioni, spesso nella pratica queste non risultano utili per modellare il sistema d'interesse.
- Si utilizzano le T-norme come condizionali (Sistemi di Mamdani), e in particolare le operazioni min e prod.
- Es: IF x is A THEN y is B

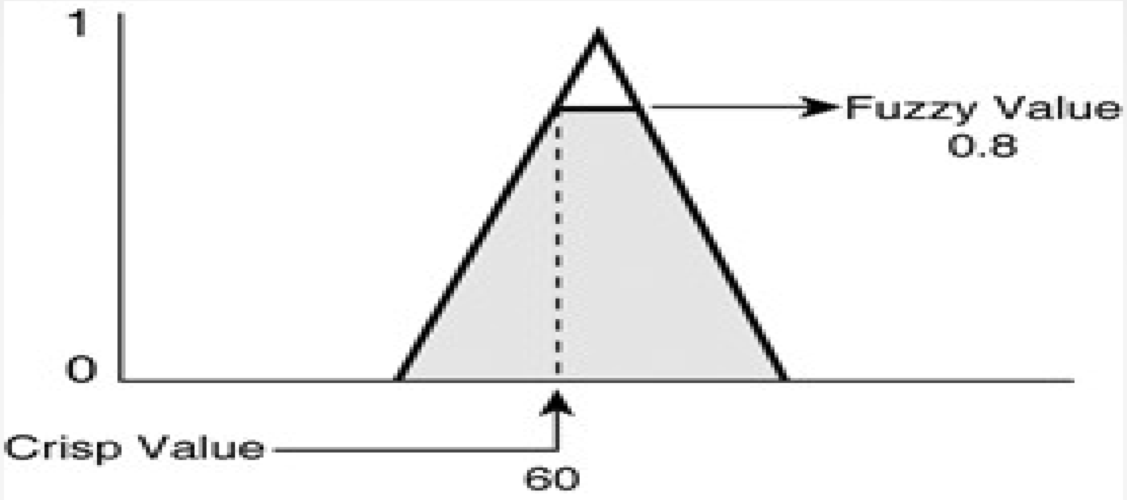
viene tradotto in $\min\{\mu_A(x), \mu_B(y)\}$

Sistema d'inferenza di Mamdani



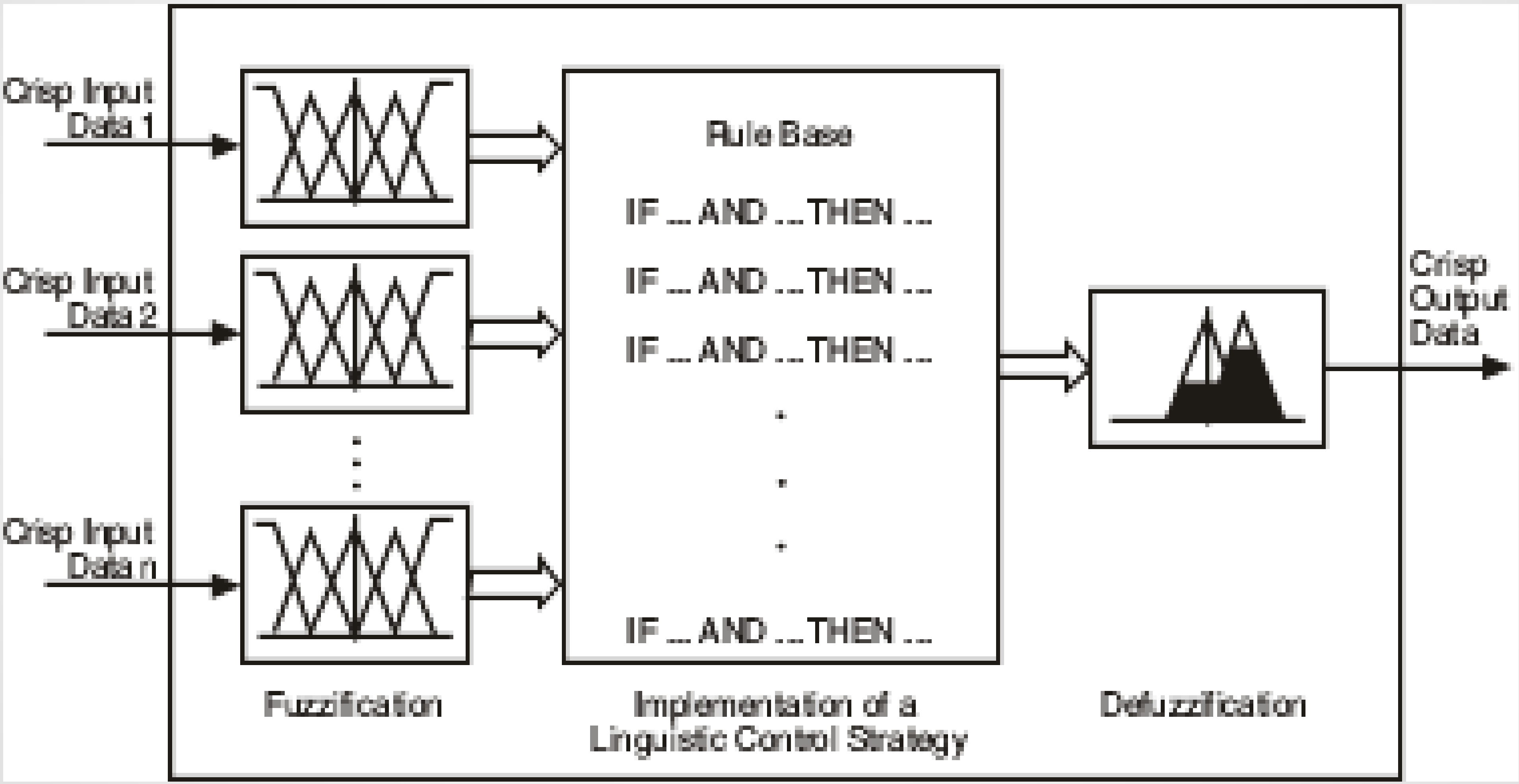
- Principalmente usato nei sistemi di controllo
- Permette di effettuare inferenze tramite regole fuzzy
- L'input e l'output sono entrambi crisp.
- I moduli “fuzzifier” e “defuzzifier” effettuano conversioni da crisp a fuzzy e viceversa.

Fuzzificazione

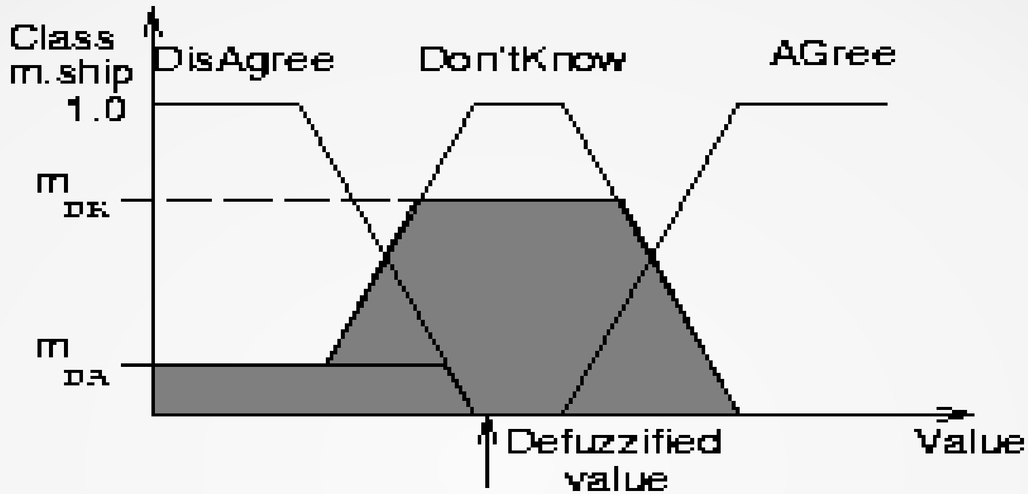


- La fuzzificazione consiste nel trasformare l'input in un valore di appartenenza.
- Si realizza calcolando il valore della funzione d'appartenenza del valore in input all'insieme fuzzy dell'antecedente di ciascuna regola.

Schema di un FIS



Defuzzificazione



- Alla fine del processo di inferenza bisogna avere un valore singolo da comunicare all'esterno.
- La defuzzificazione trasforma l'insieme fuzzy ottenuto in un valore appartenente al suo dominio.

Vediamo un'applicazione

- https://www.youtube.com/watch?v=YjL7Vgizz6U
- http://www.markbowers.org/more-projects#/levitating-ball/

Fuzzy State Machine

Fuzzy State Machines

- Gli automi a stati finiti sono un buon modello per l'AI nei videogiochi, in quanto offrono insieme semplicità di implementazione e performance elevate.
- Il loro limite consiste negli stati discreti che rendono il comportamento dei personaggi prevedibile e poco naturale
- Gli automi a stati fuzzy (FuSM) permettono di migliorare entrambe le situazioni perdendo poco in semplicità di implementazione e velocità di calcolo.

Due estensioni Fuzzy

- **STATI FUZZY (FuSM):**

  L'automa può trovarsi in più di uno stato con diversi gradi per ciascuno. Utile per modellare il comportamento o le emozioni dei personaggi.

- **TRANSIONI FUZZY:**

  L'automa si trova in un solo stato alla volta, ma gli eventi innescano più transizioni. Di queste può esserne scelta una sola in base ai pesi assegnati o un fattore di casualità. Permette azioni impreviste e apprendimento.

Perchè usare FuSM in un gioco?

“FuSMs increase gameplay by providing for more interesting responses by NPCs, by enabling less predictable NPC behavior, and by expanding the options for choices to be made by the human player.

Thus, a player does not encounter an NPC that is just MAD or not MAD about being attacked by the player. Instead, the player must deal with an NPC that can be various degrees of being MAD. This broader array of considerations increases gameplay by adding to the level of responses that can be developed for the NPC, and seen by the human player.

Another effect of adding FuSMs to computer games is to increase the replayability of the game. By broadening the range of responses and conditions that the player may encounter in given situations during the game, the player will be more likely to experience different outcomes in similar situations.”

- Eric Dybsand, Game Programming Gems 2

FuSM - Idea

- Sono una generalizzazione degli FSA in cui l'appartenenza ad uno stato è fuzzy.
- Le transizioni sono rappresentate da regole fuzzy.
- Le transizioni ricevono un grado d'appartenenza.
- Una FuSM può trovarsi in più di uno stato in un dato momento, con diversi gradi per ciascuno stato.

FuSM – Transizione 1

- Sia $S = \{S_0, S_1, \dots , S_n\}$ l'insieme degli stati.
- $\mu(S_i)$ è il grado d'appartenenza allo stato S_i e ne rappresenta il peso al tempo τ.
- Le transizioni sono rappresentate da regole del tipo if *x* is A then next state is S_i.
- Sia $t_i(j)$ una transizione da S_i a S_j, e $\mu(t_i(j))$ il suo grado d'appartenza al tempo τ.
- Sia $\mu'(S_i)$ il grado d'appartenenza di S_i al tempo $\tau + 1$.

FuSM – Transizione 2

- Il contributo di $t_i(j)$ al nuovo stato S_j è dato da

  $\mu(S_i)\mu(t_i(j))$.

- Bisogna sommare i contributi delle transizioni da tutti gli stati S_i con $\mu(S_i) > 0$ per ottenere il nuovo grado d'appartenenza a ciascuno stato $\mu'(S_j)$.

- I nuovi gradi d'appartenenza sono dati da:

$$\mu'(S_j) = \sum_{t_i(j) \in T} \mu(S_i)\mu(t_i(j))$$

FuSM - Problemi

- Considerando che $\mu(S_i) \leq 1$ e $\mu(t_i(j)) \leq 1$, è possibile che la FuSM degeneri producendo:

$$\sum_{i=1}^{n} \mu(Si) \rightarrow 0$$

- E' anche possibile che in qualche momento

$$\sum_{i=1}^{n} \mu(Si) > 1.0$$

FuSM - Restrizioni

- Per evitare la degenerazione di un FuSM bisogna imporre due vincoli:

  1) La somma dei gradi d'appartenenza di tutti gli stati deve essere pari ad uno in qualsiasi istante di tempo.

$$\sum_{i=1}^{n} \mu(Si) = 1$$

  2) La somma dei gradi d'appartenenza di tutte le transizioni che lasciano uno stato deve essere pari ad uno.

$$\sum_{t_i(x) \in T_i} \mu(t_i(x)) = 1$$

FuSM - Soluzioni

- Cosa fare se la somma delle transizioni è maggiore di 1.0? Normalizzazione:

$$\mu(t_i(j)) = \frac{\mu(t_i(j))}{\sum_{t \in T_i} \mu(t)}$$

- Che fare se è < 1.0? La differenza deve restare nello stato, in modo simile ad una transizione verso se stesso e questo contributo sarà pari a:

$$\mu(S_i) \cdot (1.0 - \sum_{t_i(j) \in T_i} \mu(t_i(x)))$$

PRO e CONTRO

Vantaggi

- Semplicità del modello e dell'implementazione
- Computazionalmente semplice
- Regole non sequenziali
- Utilizzabili per l'apprendimento
- Utili anche per modelli non-lineari

Svantaggi

- Molti agenti richiedono centinaia di regole da valutare
- Necessita di una knowledge base
- Esplosione combinatoria

FuSM e videogiochi

Le FuSM sono state usate nei seguenti giochi:

- S.W.A.T. 2: personalità dei personaggi
- Civilization: Call to Power: priorità strategiche
- Enemy Nations: modello dell'AI
- The Sims: personalità dei personaggi e interazione con oggetti.

Michele Pirovano, *The use of Fuzzy Logic for Artificial Intelligence in Games, 2012.*

GROUP DECISION MAKING

GDM – Cos'è?

- E' il processo cognitivo di prendere le decisioni all'interno di un gruppo.
- Argomento multidisciplinare.
- Applicazioni in informatica e intelligenza artificiale.
- Campo molto vasto, noi vedremo solo un'applicazione che riguarda il “soft consensus”.

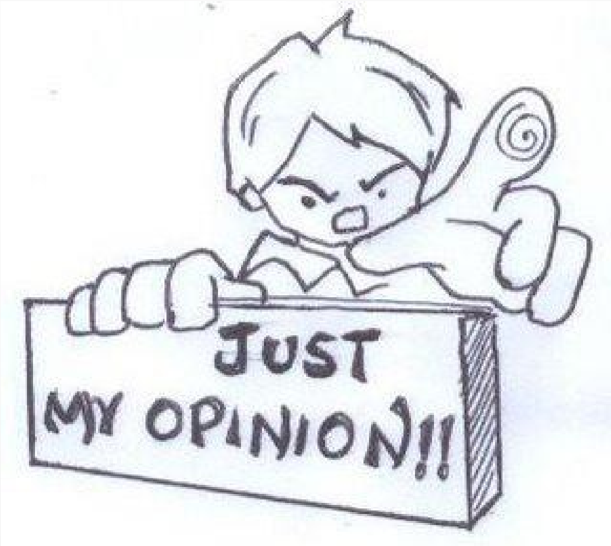
"OK, all those in favour of delegating decision-making, shrug your shoulders"

Soft Consensus

- Consenso non unanime tra i decision-makers.
- Le preferenze tra le varie alternative sono espresse in modo graduale.
- L'informazione è spesso incompleta.
- Vengono mutuate tecniche dalla logica fuzzy.

Opinione di maggioranza

- Il concetto di soft-consensus è legato a quello di opinione di maggioranza.
- Ci sono vari metodi per trovare un valore rappresentativo per il gruppo.
- La tecnica usata e il significato che si dà a “maggioranza” influiscono molto sul risultato.

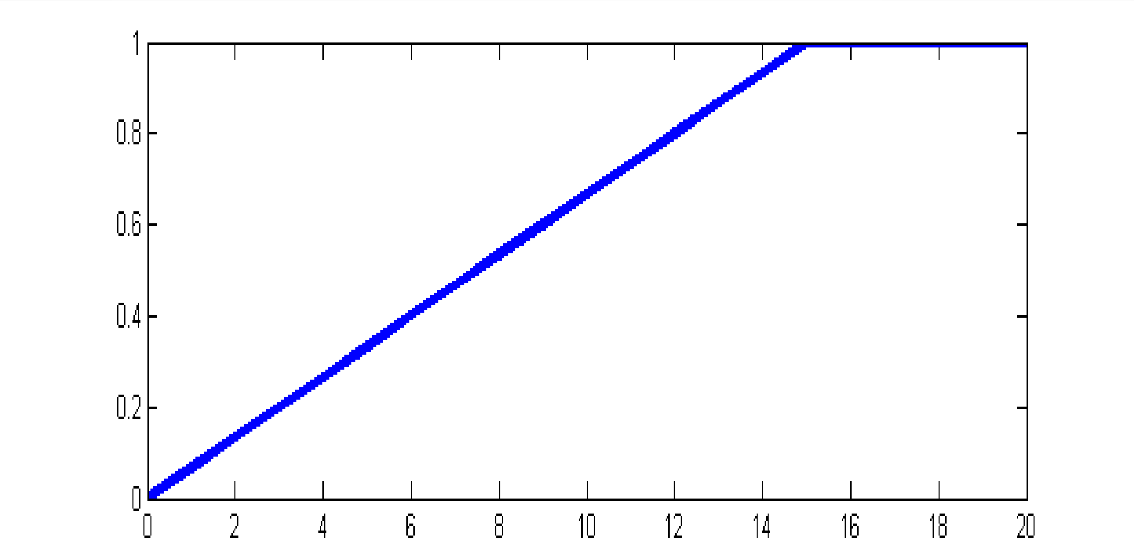


Ordered Weighted Average (OWA)

- Sia $W=[w_1, w_2, ..., w_n]$ un vettore pesi tale che la somma sia 1, e sia $A=[a_1, a_2, \ldots, a_n]$ un vettore di valori nell'intervallo [0, 1] che indicano i valori di preferenza di n decisori.
- $OWA(a_1, a_2, \ldots, a_n) = \Sigma b_i w_i$
  dove b_i è l'i-esimo elemento più grande a_j
- Ha bisogno di un quantificatore linguistico Q per calcolare i pesi.
- Il risultato è “la percentuale di criteri soddisfatti”.

Quantificatori linguistici

- Estensione dei quantificatori della logica classica.
- Permettono di riferirsi a un numero variabile di elementi di un insieme.
- Sono accompagnati da un'etichetta linguistica come “la maggior parte”, “qualche”, “più o meno k”.



Calcolo dei pesi

- I pesi sono il parametro che determina la semantica dell'operatore.
- Vengono calcolati a partire dal quantificatore linguistico associato all'OWA.

$$w_i = Q\left(i/n\right) - Q\left(\left(i-1\right)/n\right)$$

Esempio

- Abbiamo 6 decisori con il quantificatore linguistico qua a destra.
- Le opinioni espresse sono:

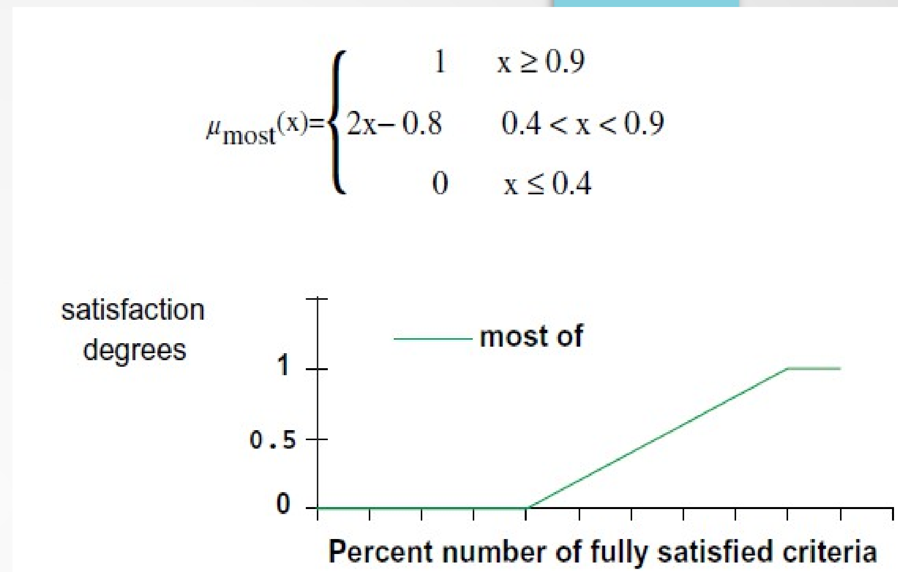
  [0.5, 0.3, 0.6, 1, 1, 0.1]

- Si ordinano le opinioni in ordine decrescente:

  B=[1, 1, 0.6, 0.5, 0.3, 0.1]

- Si calcolano i pesi:

  W=[0, 0, 0.2, 0.33, 0.33, 0.14]



- Il prodotto scalare <B,W> produce 0.3980.
- Sembra un valore ragionevole?

Induced Order WA (IOWA)

- Introduciamo un vettore *I*=[i_1, i_2, …, i_n] di *induzione.* I valori sono adesso delle coppie (i_j, a_j).
- IOWA((i_1,a_1), (i_2,a_2), …, (i_n,a_n)) = $\Sigma b_i w_i$
  dove b_i è l'elemento a_j accoppiato all'i-esimo più grande i_j.
- Possiamo trovare un algoritmo che calcoli un vettore di induzione che ci permetta di ottenere un risultato espressione della maggioranza dei valori simili?

Algoritmo

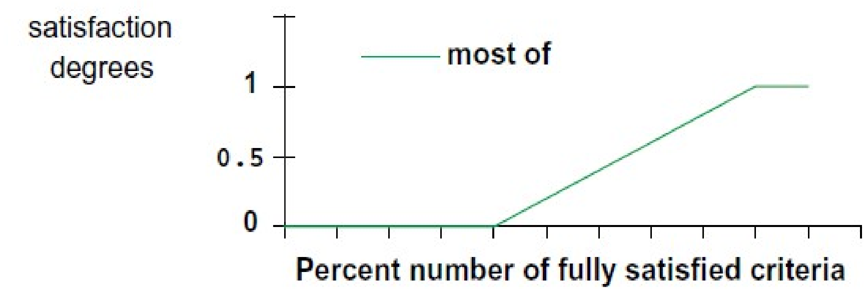
- Scegliamo un valore α di "tolleranza".
- Per ciascun valore calcoliamo un supporto s_i, che conta il numero di valori simili a a_i, cioè quelli che si trovano a una distanza inferiore ad α.
- $I=[s_1, s_2, \ldots, s_n]$ è il vettore di induzione.
- Il vettore dei pesi W si calcola come prima.
- Dopo aver ottenuto B dall'ordinamento di A secondo *I*, calcoliamo il prodotto scalare <B, W>.

Esempio

- Scegliamo α=0.3
- A=[0.5, 0.3, 0.6, 1, 1, 0.1]
- I=[2, 3, 2, 1, 1, 1]
- Si ordinano le opinioni in ordine crescente di I:

  B=[0.1, 1, 1, 0.5, 0.6, 0.3]
- W=[0, 0, 0.2, 0.33, 0.33, 0.14] come prima.
- Il prodotto scalare <B, W> produce 0.6050.
- Il risultato è maggiormente influenzato dal “cluster” di valori simili [0.3, 0.5, 0.6].

$$\mu_{most}(x)=\begin{cases} 1 & x \geq 0.9 \\ 2x-0.8 & 0.4 < x < 0.9 \\ 0 & x \leq 0.4 \end{cases}$$



- E' possibile pensare a tecniche che modificano il calcolo di W o I per ottenere risultati ancora diversi, purché rappresentino una semantica utile per il contesto.

Possibili applicazioni

- La teoria delle decisioni di gruppo non è limitata al solo caso di gruppi umani che devono prendere decisioni.
- Applicazioni nell'information retrieval (meta-search), ma può essere usata in tutti quei casi in cui sono presenti agenti software multipli che calcolano un valore in modo approssimato e bisogna decidere il valore più realistico.

Conclusioni

- La logica fuzzy può essere usata in ambiti molto diversi per gestire informazioni in condizioni di incertezza.
- Non si esce – ovviamente - dai confini del calcolabile, ma essa fornisce un framework matematico che permette di trovare soluzioni approssimate per problemi altrimenti “difficili”.
- Dalla logica fuzzy è nato un settore di ricerca, ancora in pieno sviluppo, chiamato “Soft Computing”, che permette la modellizzazione di problemi difficili attraverso tecniche euristiche o di approssimazione.

Grazie per l'attenzione