

# درس مبانی هوش محاسباتی

## پروژه امتیازی

### پروژه فازی

(یک مسئله ی کاربردی)

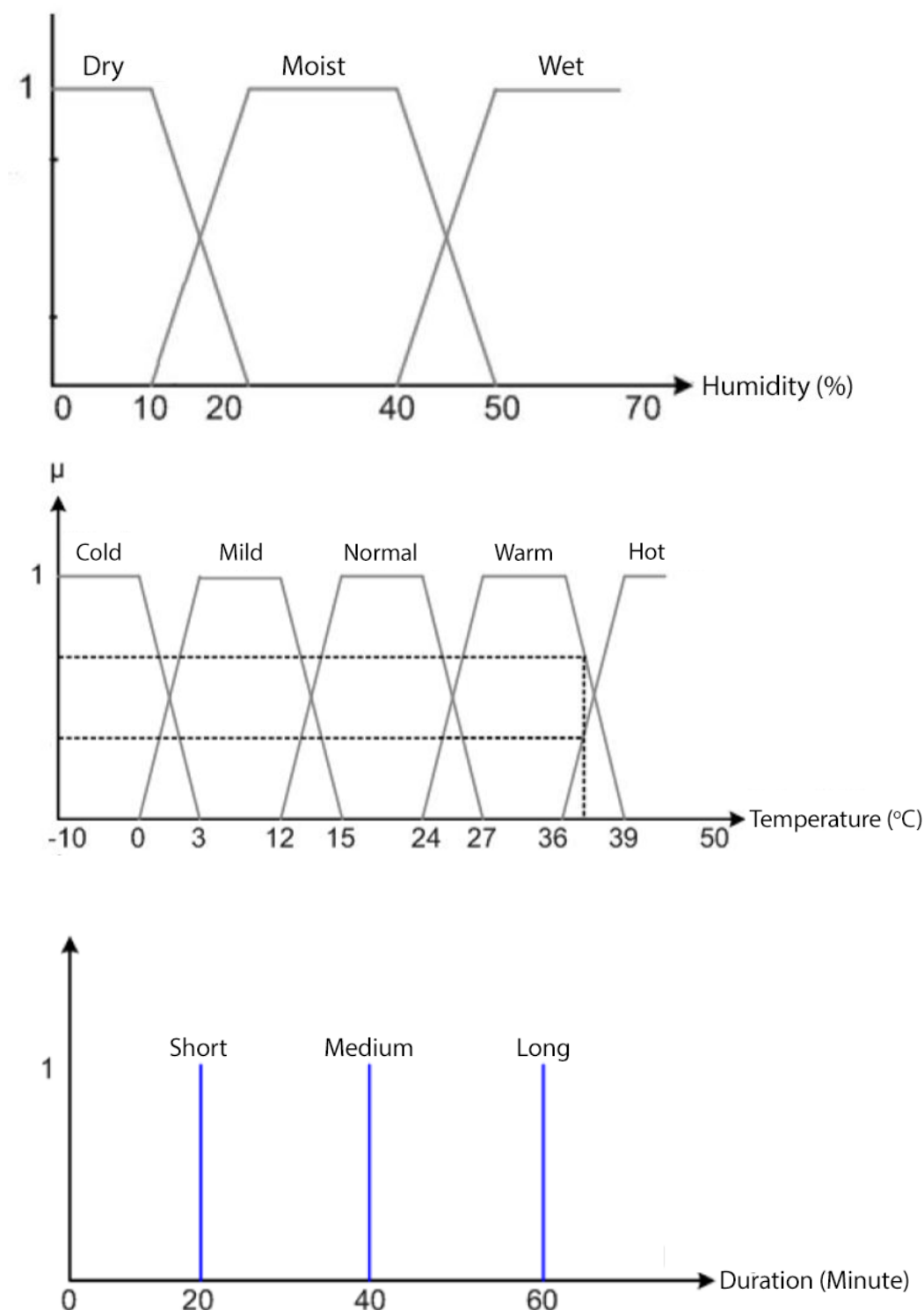
دلارام رجایی ۹۷۳۱۰۸۴



دانشگاه صنعتی امیرکبیر  
( پلی تکنیک تهران )

## مسئله‌ی آبیاری خاک (Fuzzy-System-for-Sprinkle)

در این مسئله ما میخواهیم با توجه به دو ورودی دما و رطوبت خاک، مدت زمان آبیاری خاک را مشخص کنیم. برای حل مسئله به کمک منطق فازی، لازم است مقادیر ما از حالت مطلق به حالت فازی (نادقیق، نسبی) تبدیل شوند. به این مرحله Fuzzification یا فازی سازی گفته می شود. برای این منظور می‌بایست مجموعه های فازی تعریف شود و طبق تابع تعلق، میزان تعلق هر مقدار به هر مجموعه محاسبه شود. برای اینکار به کمک نقاط داده شده معادلات خطی soil moisture و temperature ، duration را می‌نویسیم.



سپس به کمک این معادلات مقدار تابع تعلق soil moisture و temperature را بدست می‌آوریم. ابتدا تابع fuzzification صدا زده می‌شود که در آن دو تابع زیر صدا زده می‌شود:

```
self.fuzzify_soil_moisture(input['soil_moisture'])
self.fuzzify_temperature(input['temperature'])
```

در هر کدام از این توابع معادلات آنها صدا زده می‌شود و با توجه به بازه‌ای که نقطه در آن قرار دارد در معادله‌ی مربوطه قرار گرفته و مقداری برگردانده می‌شود.

معادلات در فایل پایتون دیگری به نام equations قرار دارند.

```
def fuzzify_soil_moisture(self, input):
    soil_moisture = Equations.soil_moisture()
    self.membership_soil_moisture["dry"] = soil_moisture.dry(input)
    self.membership_soil_moisture["moist"] = soil_moisture.moist(input)
    self.membership_soil_moisture["wet"] = soil_moisture.wet(input)

def fuzzify_temperature(self, input):
    temperature = Equations.temperature()
    self.membership_temperature["cold"] = temperature.cold(input)
    self.membership_temperature["mild"] = temperature.mild(input)
    self.membership_temperature["normal"] = temperature.normal(input)
    self.membership_temperature["warm"] = temperature.warm(input)
    self.membership_temperature["hot"] = temperature.hot(input)
```

در مرحله بعد لازم است مقادیر فازی بدست آمده در قوانین موجود برای حل مسئله بررسی شوند. به این مرحله Inference گفته می شود. ابتدا تابع `self.inference()` صدا زده می شود و در آن قوانین چک میشوند:

```
def inference(self):
    self.duration["long"] = max(
        min(self.membership_soil_moisture["dry"],
self.membership_temperature["cold"]),
        min(self.membership_soil_moisture["dry"],
self.membership_temperature["mild"]),
        min(self.membership_soil_moisture["dry"],
self.membership_temperature["normal"]),
        min(self.membership_soil_moisture["dry"],
self.membership_temperature["warm"]),
        min(self.membership_soil_moisture["dry"],
self.membership_temperature["hot"]))
    )
    self.duration["medium"] = max(
        min(self.membership_soil_moisture["moist"],
self.membership_temperature["normal"]),
        min(self.membership_soil_moisture["moist"],
self.membership_temperature["warm"]),
        min(self.membership_soil_moisture["moist"],
self.membership_temperature["hot"]),
    )
    self.duration["short"] = max(
        min(self.membership_soil_moisture["moist"],
self.membership_temperature["cold"]),
        min(self.membership_soil_moisture["moist"],
self.membership_temperature["mild"]),
        min(self.membership_soil_moisture["wet"],
self.membership_temperature["cold"]),
        min(self.membership_soil_moisture["wet"],
self.membership_temperature["mild"]),
        min(self.membership_soil_moisture["wet"],
self.membership_temperature["normal"]),
        min(self.membership_soil_moisture["wet"],
self.membership_temperature["warm"]),
        min(self.membership_soil_moisture["wet"],
self.membership_temperature["hot"]))
    )
```

قانون های نوشته شده بر حسب جدول زیر هستند:

	Cold	Mild	Normal	Warm	Hot
Dry	Long	Long	Long	Long	Long
Moist	Short	Short	Medium	Medium	Medium
Wet	Short	Short	Short	Short	Short

همانطور که دیده شد با توجه به قانونی که گفته شد مقادیر بدست آمده در مرحله ی قبل را در قوانین قرار داده هر کجا که and بود را min گرفتیم و در آخر تمامی آنهایی که خروجی یکسانی داشتند به طور مثال مربوط به short، duration می‌شدند را با هم or کردیم که همان max است.

به این ترتیب تابع self.duration ما بدست می‌آید که همان حد قدرت ما است.

مرحله آخر Defuzzification نام دارد. در این مرحله به کمک استنتاج های انجام شده، مجدد به دنیای مقادیر مطلق بر می‌گردیم تا نیر و و جواب را به صورت مقدار مطلق بدست آوریم. برای غیرفازی سازی نیز روش های مختلفی وجود دارد که از مهم ترین و پرکاربردترین آن ها روش مرکز جرم می باشد.

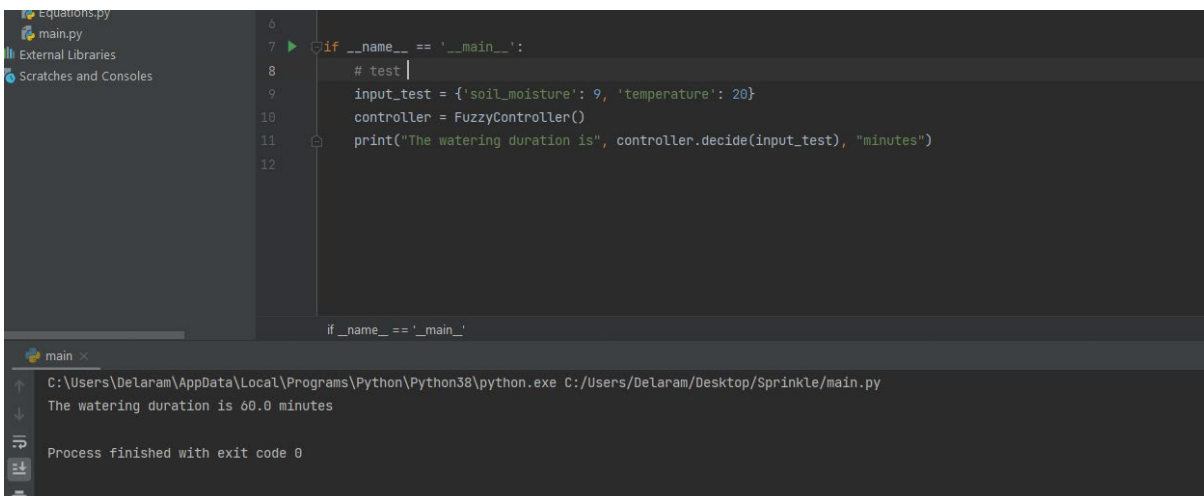
در این مرحله تابع self.defuzzification() صدا زده می‌شود و مقدار آن به عنوان duration برگردانده می‌شود.

ده هزار نقطه بین ۰ تا ۱۰۰ انتخاب می‌کنیم، سپس برای هرکدام مقدار تعلق آن نقطه محاسبه می‌شود و چک می‌شود که حتما کمتر مساوی self.duration باشد. سپس بین تمامی این مقادیر max آنها برگردانده می‌شود. سپس با توجه به فرمول زیر مرکز جرم را محاسبه می‌کنیم:

$$\frac{\sum \mu \times \Delta x \times dx}{\sum \mu \times dx}$$

مقداری که بدست می‌آید همان duration است.

نتیجه به صورت زیر قابل مشاهده است:



The screenshot shows a Python IDE with a file named `main.py` open. The code in the editor is as follows:

```

6
7 if __name__ == '__main__':
8     # test |
9     input_test = {'soil_moisture': 9, 'temperature': 20}
10    controller = FuzzyController()
11    print("The watering duration is", controller.decide(input_test), "minutes")
12

```

The output console at the bottom shows the command executed and the result:

```

C:\Users\Delaram\AppData\Local\Programs\Python\Python38\python.exe C:/Users/Delaram/Desktop/Sprinkle/main.py
The watering duration is 60.0 minutes
Process finished with exit code 0

```

برای سنجش درستی برنامه تستی داده شده است که در آن رطوبت خاک، ۹ که به معنی `dry`، دمای خاک برابر ۲۰ به معنی `normal` است که با توجه به جدول قانون ها به معنی `long` است که یعنی ۶۰ دقیقه باید آبیاری شود. که برنامه نتیجه ی درستی را به ما برگردانده است.

کدهای مربوطه شامل سه فایل پایتون `main`، `controller` و `equations` در فایل زیپ قرار دارد.