



Софийски Университет „Св. Климент Охридски“
Факултет по Математика и Информатика

Курсова работа

за курса по

Размити множества

на тема

„Определяне на активност на риба и водни организми в зависимост метеорологични и астрономически явления“

Тодор Борисов Михайлов, ф.н.24425

магистър, специалност „Извличане на Информация и Откриване на знания“

Дата 06.02.2014

Въведение

Целта на проекта е да се създаде приложение, което по известни входни данни за метеорологични условия и астрономически (разположение на луната и слънцето) да определи активността на рибите и водните организми. Информацията е изключително полезна при извършването на любителски риболов в сладководни водни басейни.

Теоретична постановка и използван алгоритъм

Доказано е, че активността на рибата и водните организми и тяхното хранене зависи от точно определени фактори. Направени са много наблюдения и изследвания за поведението на животните в зависимост от метеорологичните и астрономическите явления. Водните организми са едни от най-чувствителните към тези явления. От активността на рибата зависят пряко резултатите от уловите в любителския риболов.

Цел

В текущата курсова работа са разгледани зависимостите на тези явления и начинът по който влияят на активността на рибата в сладководните водни басейни. Зависимостите между метеорологичните и астрономическите условия и влиянието върху активността на рибата са извадени от различни източници, включващи научни теории и наблюдения. Източниците са посочени в раздел „Използвана литература“.

Фактори

Факторите, които влияят най-силно на активността на рибата и водните организми са следните:

- Метеорологични условия
 - o Температура
 - o Изменение на температурата за кратък период от време
 - o Влошаване и подобряване на времето – изменянето на атмосферното налягане за кратък период от време
- Астрономически явления
 - o Времеви интервали около изгряване и залез на слънцето
 - o Времеви интервали около изгряване и залез на луната

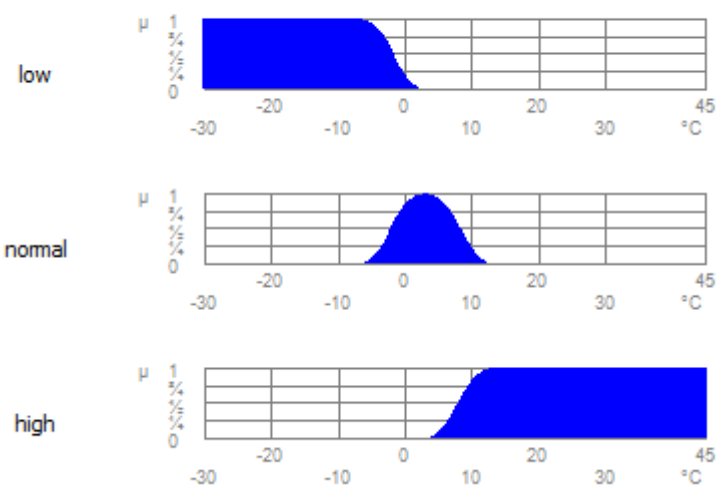
За да представим тези фактори, ще използваме размити множества, които ги описват най-близо. Ще изградим зависимостите между тези явления и активността на рибата и ще покажем как си взаимодействат едновременно.

Влияещите на активността фактори ще бъдат представени чрез размити множества:

Факторите са:

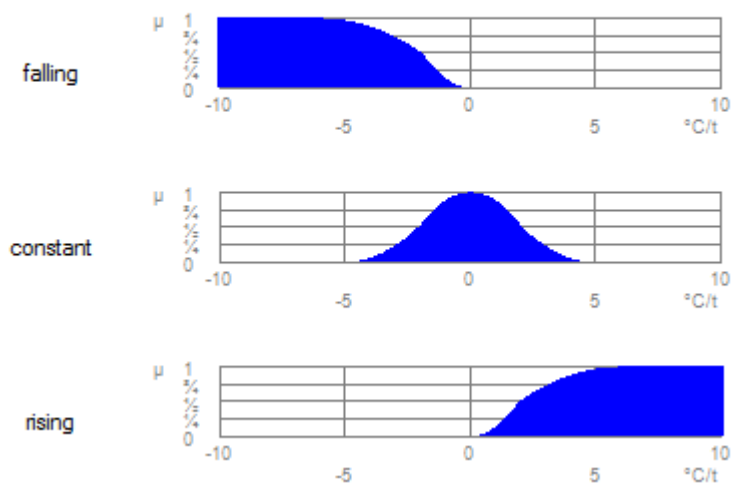
1. Temperature - Температура за сезона разделена на:
 - a. lowTemperature – Ниска температура
 - b. goodTemperature – Нормална температура
 - c. highTemperature – Висока температура
- Температурата е представена в градуси по целзии

Temperature **ΔTemperature** Weather Change Sun/Moon peaks Fish Activity

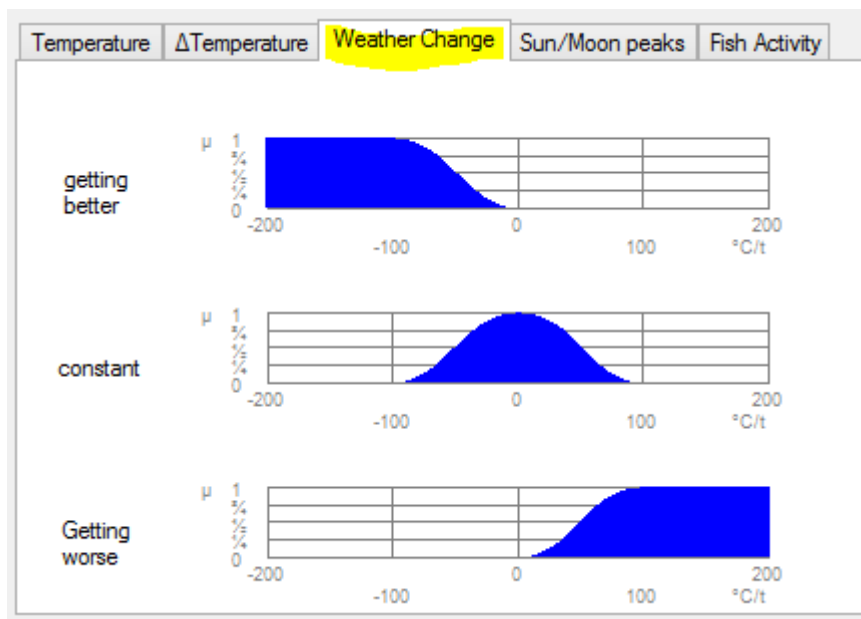


2. Delta Temperature – изменение на температурата за определен период от време.
 - a. risingTemperature – температурата се покачва;
 - b. fallingTemperature – температурата пада;
 - c. constantTemperature – температурата е постоянна;
- Изменението на температурата е представена в градуси по целзии

Temperature **ΔTemperature** Weather Change Sun/Moon peaks Fish Activity

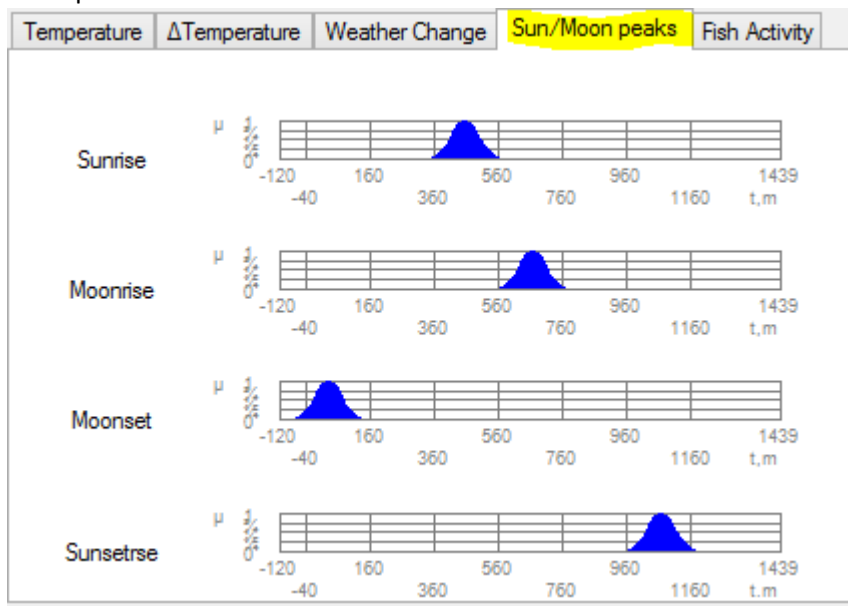


3. Промяна на времето – представено е чрез атмосферното наляне и неговото изменение за малък период от време (30 мин)
 - a. weatherGettingBetter – времето се подобрява;
 - b. weatherNoChange – времето не се променя;
 - c. weatherGettingWorse – времето се „разваля“;
- Изменението на налягането на въздуха е представено в Нр.



4. Състояние на слънцето и луната

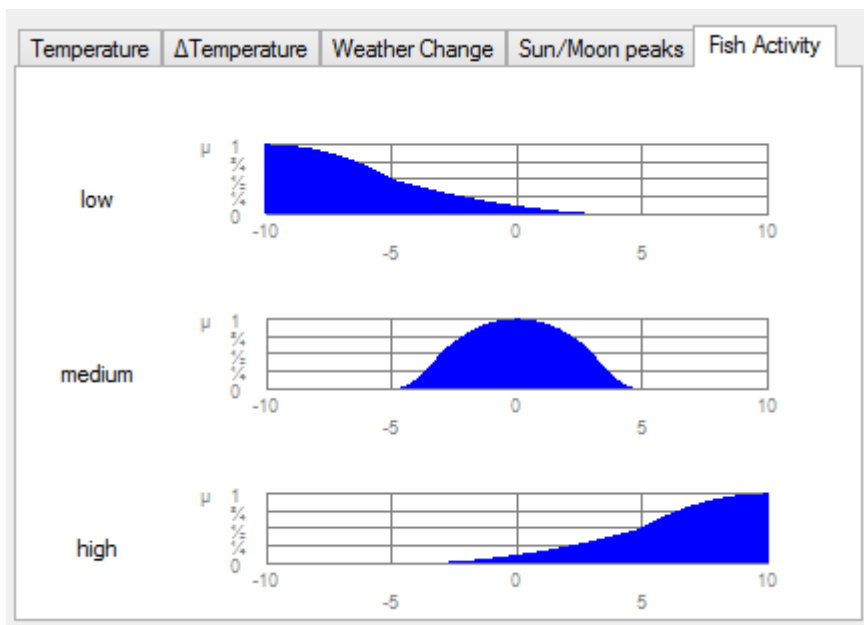
- moonRise – Изгрев на Луната;
 - moonSet – Залез на Луната;
 - sunRise – Изгрев на Слънцето;
 - sunSet – Залез на Слънцето;
- Периодите на изгрев и залез на Слънцето и Луната са представени във времеви интервал.



Последствието от изменението на факторите е активността на рибата.

Активността на рибата – представена е чрез степен на активност от -10 до 10 (след дефъзификацията)

- lowActivity – ниска активност;
- mediumActivity; - средна активност
- highActivity; - висока активност



Входни данни

Входните данни ще бъдат:

- Дата и час – за определяне на периодите на изгрев/залез на слънцето и луната (в момента периодите за изгрев/залез се задават ръчно като входни параметри.)
- Температурата – температурата в дадения дата и час;
- Изменение на температурата – за последните 3 часа;
- Изменение на атмосферното налягане за последните 3 часа ;

Информацията за метеорологичните условия и времето, в което слънцето и луната залязват ще се вземат от уеб услуга, която предоставя информация за времето 5 дни назад във времето и 5 дни напред през 3 часа.

Използвани оператори

За да бъде предвидена активността на рибата в зависимост от факторите ще използваме if-then клаузи от условията, които влияят и активността при съответните условия. За целта ще използваме Mamdani Implication. За сумата на всички правила ще използваме Root Square Sum, а за дефъзификация Center of Gravity.

Root-square-sum е метод, който се използва за по-точно сумиране на резултати в размити множества.

Mamdani Implication е метод за импликация, който е базиран на приетото правило, че максималната стойност на функцията на принадлежност на резултатът от логическия извод не може да бъде по-голяма от функцията на принадлежност на условията. Това означава, че графиката на функция на резултатът е отрязък, който не надхвърля стойностите на условието.

Правила и изводи

Правилата на зависимост между входните условия и последствията са:

Ако lowTemperature **и** risingTemperature **Тогава** highActivity

Ако weatherGettingWorse **Тогава** highActivity

Ако weatherGettingBetter **Тогава** highActivity

Ако moonRise **или** moonSet **или** sunRise **или** sunSet **Тогава** highActivity

Ако goodTemperature **и** risingTemperature **Това** lowActivity
Ако goodTemperature **и** fallingTemperature **Това** lowActivity

Ако goodTemperature **и** constantTemperature **Това** mediumActivity
Ако (**не** moonRise) **и** (**не** moonSet) **и** (**не** sunRise) **и** (**не** sunSet) **Това** mediumActivity

Правилата, представени в код

За реализация на проекта е използван Fuzzy Framework за C#. Правилата по-горе представени като програмен код са:

```
var Result = ((lowTemperature&risingTemperature)&highActivity)%  
((weatherGettingWorse)&highActivity)  
%((weatherGettingBetter)&highActivity)%  
((moonRise|moonSet|sunRise|sunSet)&highActivity)%  
((goodTemperature&risingTemperature)& lowActivity)%  
((goodTemperature&fallingTemperature)&lowActivity)%  
  
((goodTemperature&constantTemperature)&mediumActivity)%  
(!moonRise&!moonSet&!sunRise&!sunSet)&mediumActivity);
```

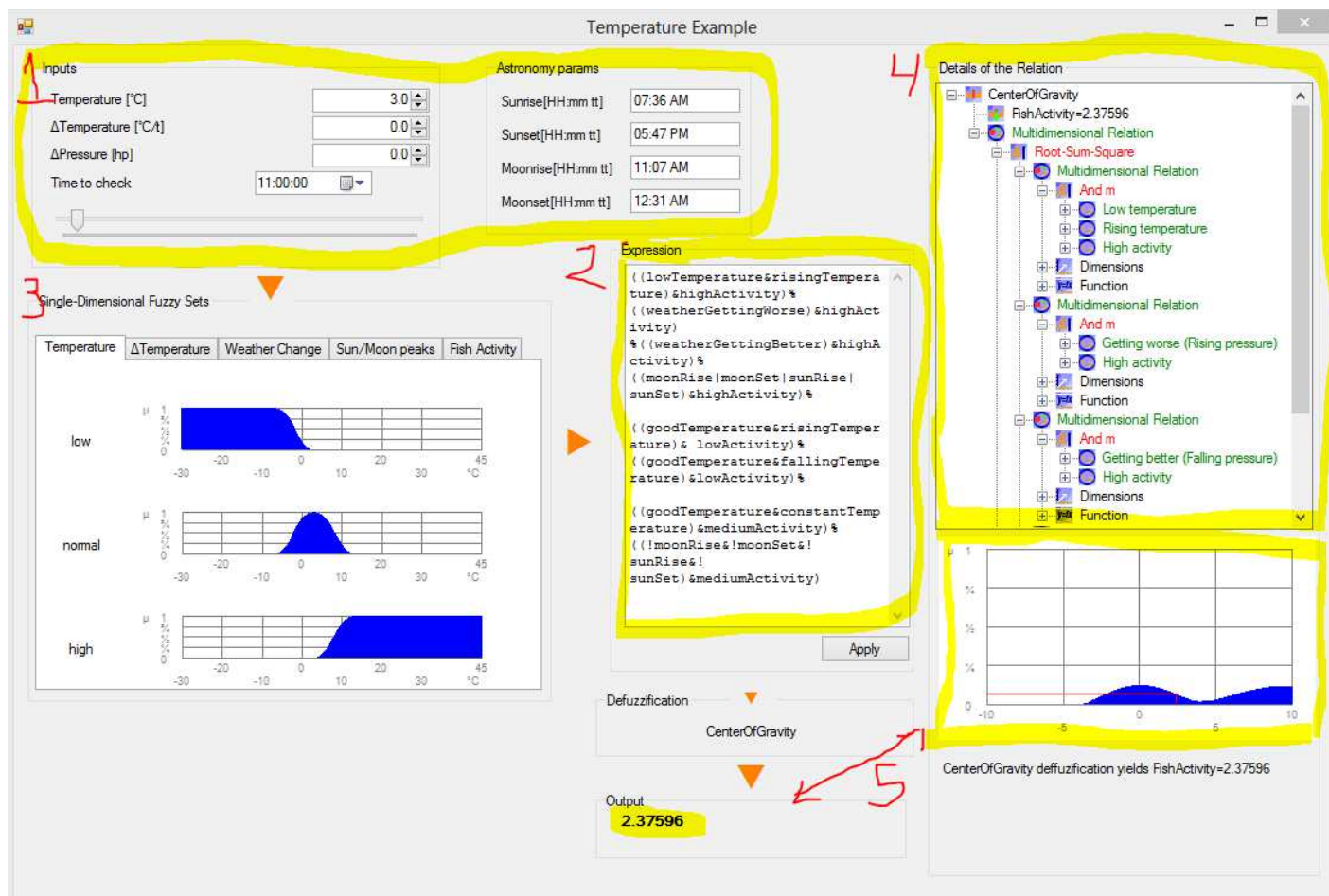
Използвани оператори за работа с размити множества във Fuzzy Framework

- & - при използване на този оператор между стойности от входните данни, той се интерпретира като обединение на 2-те fuzzy стойности ($\min(\mu_A(x), \mu_B(x)), x \in U$)
- & - При използване на операторът & между условията и последствието той се интерпретира от Framework-а като Mamdani импликация.
- % - интерпретира се като Root Sum Square между стойности от размити множества.
- ! – интерпретира се като complement ($1 - \mu_A(x), x \in U$)

Реализирано приложение

За работа с дадените условия, входящи данни и визуализация на данните е реализирано приложение на C#.

Приложението има потребителски интерфейс, показан на фигурата по-долу:



Той има следните елементи :

- Входни данни – входните данни, които се задават са:
 - Часа, който е в момента
 - Температурата в момента
 - Изменението на температурата за последните 3 часа
 - Изменението на атмосферното налягане за последните 3 часа
 - Параметри за точен минута и час на фазите на луната и слънцето
- Правила за зависимост между условията и резултата – за по-лесни тестове може, условията могат да се сменят runtime, като правилата се компилират и интерпретират динамично от Fuzzy Framework
- Визуализация на Fuzzy стойностите
- Визуализация на релациите от зададените правила
- Визуализация на резултата като fuzzy графика и резултатът от дефъзификация

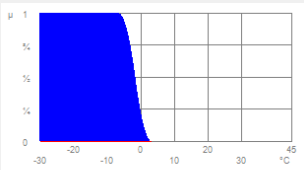
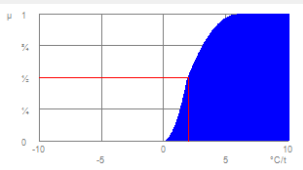
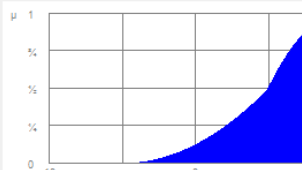
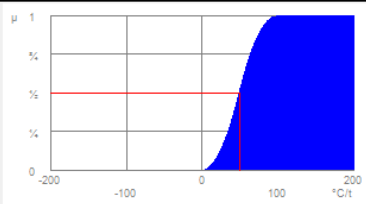
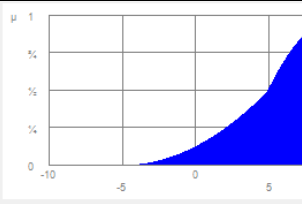
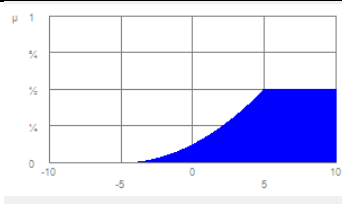
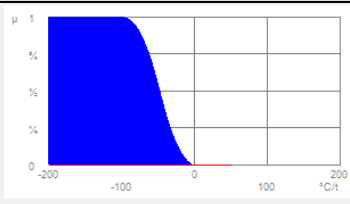
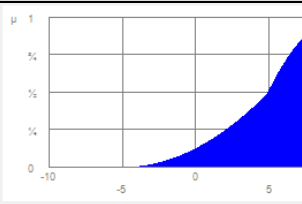
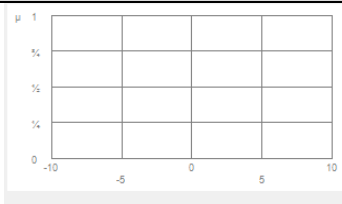
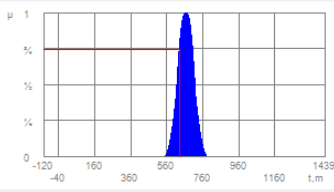
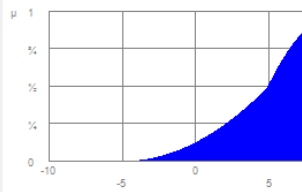
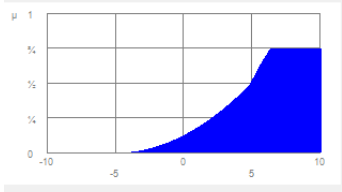
Експериментални/симулационни резултати:

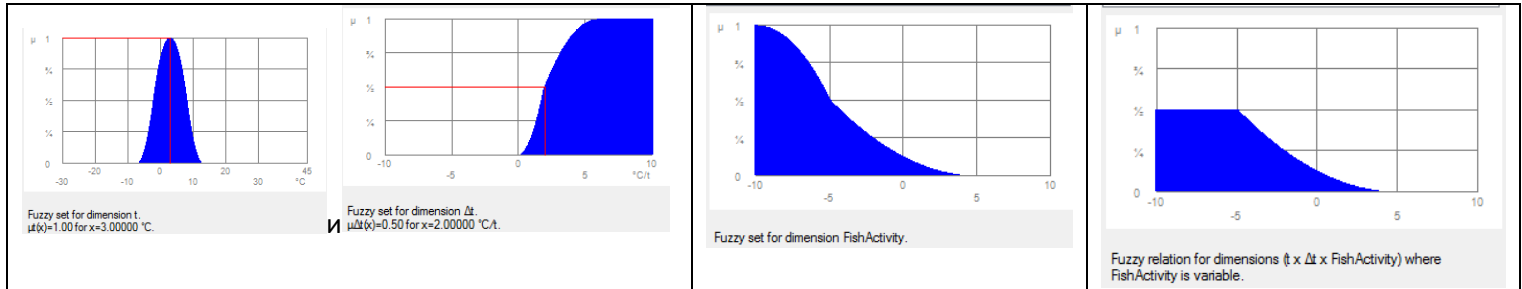
Примерни резултати от използването на приложението:

Входни данни:

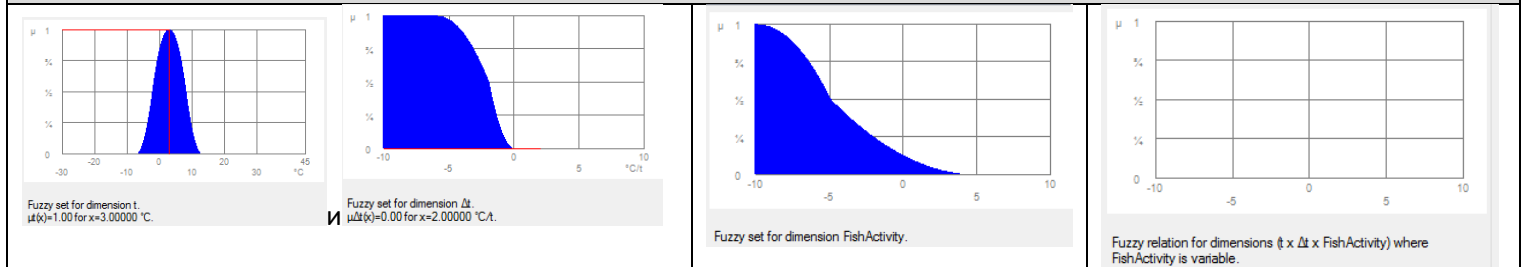
Inputs	Astronomy params
Temperature [°C]	Sunrise[HH:mm tt] 07:36 AM
ΔTemperature [°C/Δ]	Sunset[HH:mm tt] 05:47 PM
ΔPressure [hp]	Moonrise[HH:mm tt] 11:07 AM
Time to check	Moonset[HH:mm tt] 12:31 AM
10:35:00	

Резултати от правилата:

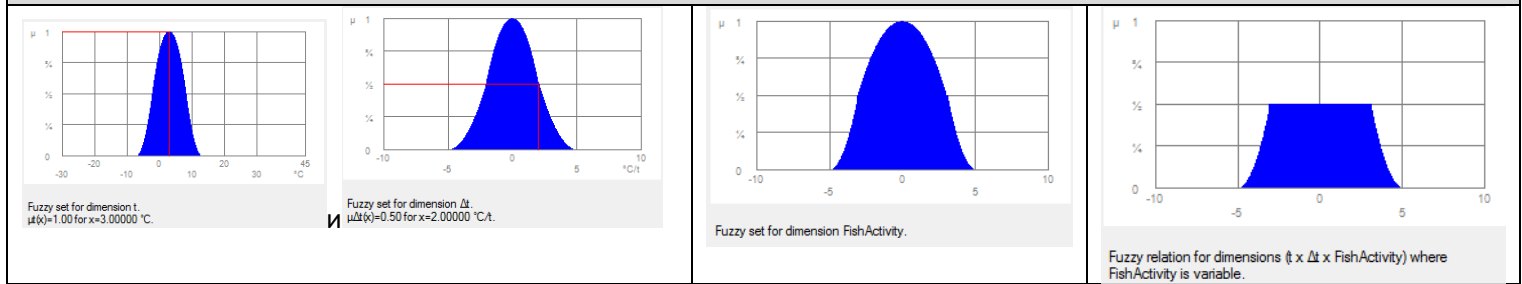
АКО	Тогава	Резултат
Ако lowTemperature и risingTemperature Тогава highActivity		
 Fuzzy set for dimension t. $\mu_t(x)=0.00$ for $x=3.00000$ °C.	 Fuzzy set for dimension Δt. $\mu_{\Delta t}(x)=0.50$ for $x=2.00000$ °C/Δ.	 Fuzzy set for dimension FishActivity.
Ако weatherGettingWorse Тогава highActivity - Има повишаване на атмосферното налягане – времето се влошава, което е се отразява положително на активността на рибата:		
 Fuzzy set for dimension Δt. $\mu_{\Delta t}(x)=0.50$ for $x=50.00000$ °C/Δ.	 Fuzzy set for dimension FishActivity.	 Fuzzy relation for dimensions (Δt x FishActivity) where FishActivity is variable.
Ако weatherGettingBetter Тогава highActivity		
 Fuzzy set for dimension Δt. $\mu_{\Delta t}(x)=0.00$ for $x=50.00000$ °C/Δ.	 Fuzzy set for dimension FishActivity.	 Fuzzy relation for dimensions (Δt x FishActivity) where FishActivity is variable.
Ако moonRise или moonSet или sunRise или sunSet Тогава highActivity		
- Часът е 10:35, което е близо до 11:07, когато е пиков момент в активността на рибата, заради „изгрева“ на луната		
 Fuzzy set for dimension time. $\mu_{time}(x)=0.75$ for $x=635.00000$ t.m.	 Fuzzy set for dimension FishActivity.	 Fuzzy relation for dimensions (time x FishActivity) where FishActivity is variable.
Ако goodTemperature и risingTemperature Тогава lowActivity		



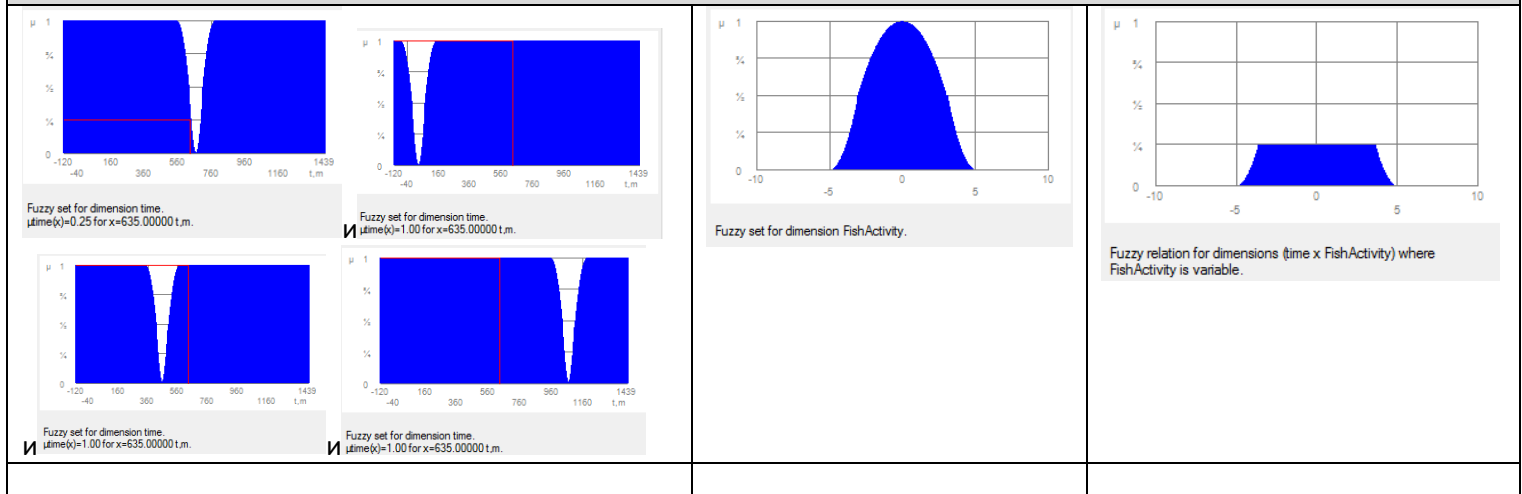
Ако goodTemperature **и** fallingTemperature **Торава** lowActivity



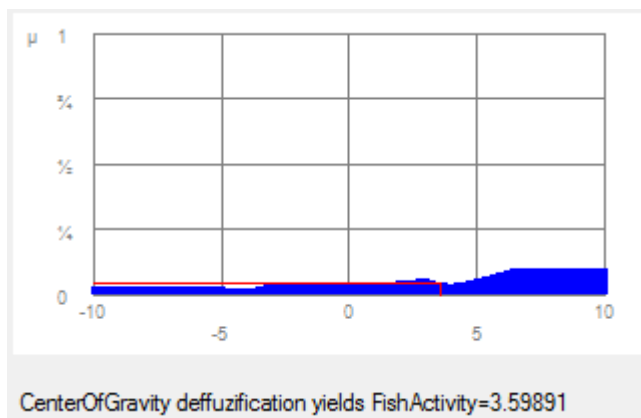
Ако goodTemperature **и** constantTemperature **Торава** mediumActivity



Ако (не moonRise) **и** (не moonSet) **и** (не sunRise) **и** (не sunSet) **Торава** mediumActivity



Краен резултат в този пример:



При дефъзификация чрез методът Center of Gravity => Активността на рибата при тези условия е 3.59891/(-10 до 10)

Заклучение:

На базата на дефинирани ясни правила за зависимостта между метеорологични и астрономически условия и активността на рибата, заедно с истински входни данни и размити множества, изчислихме потенциалната активност на рибата за дадената дата и час. Това изчисление би било много трудно на ум/на ръка от едно лице което извършва риболов.

Размитите множества намериха добра реализация в този пример.

В бъдеще от приложението, което разработихме в този курс ще бъде създадено мобилно приложение, което да служи като съветник при риболовни излети.

Литература:

http://en.wikipedia.org/wiki/Solunar_theory - Теория за зависимост на активността на животните и положението на луната и слънцето спрямо земята;

<http://myfwc.com/fishing/freshwater/fishing-tips/solunar-theory/> - съвети за подобряване на резултатите от риболов, в зависимост от положението на луната;

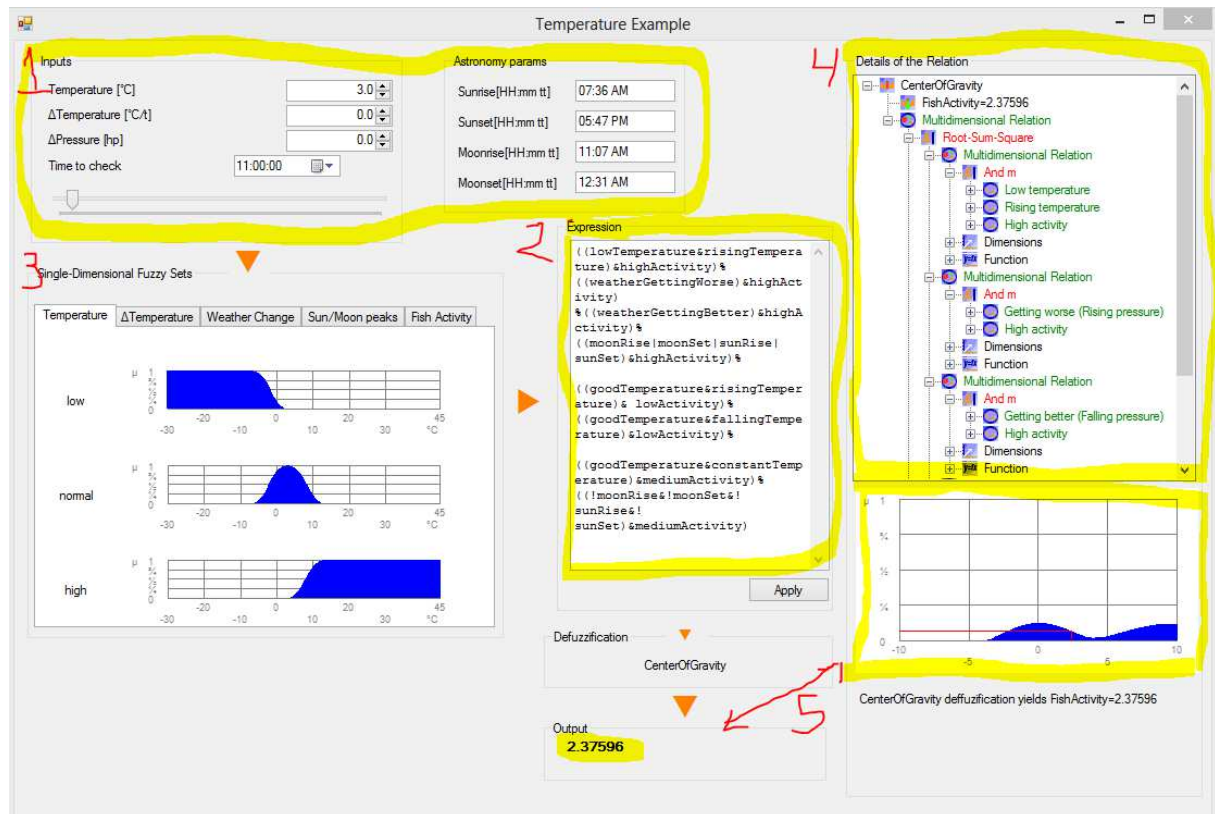
<http://www.codeproject.com/Articles/151161/Fuzzy-Framework> - Fuzzy framework - фреймуърк за работа с размити множества през C#

Приложения:

Код на програмната реализация, реализиран интерфейс, фигури и таблици.

Потребителски интерфейс

Потребителския интерфейс на приложението изглежда по следния начин.



Той е описан – по горе в документа.

Код за реализирането на зависимостите в размитите множества

Кодът приложен по-долу може да бъде видян и в предадения код на проекта

Дефиниране на Fuzzy стойностите

#region Definition of dimensions

```
protected static IContinuousDimension temperature;
protected static IContinuousDimension deltaTemperature;
protected static IContinuousDimension deltaPressure;
protected static IContinuousDimension dayTime;
protected static IContinuousDimension fishActivity;
#endregion
```

//WEATHER

//for temp:

```
public static ContinuousSet lowTemperature;
public static ContinuousSet highTemperature;
public static ContinuousSet goodTemperature;
```

//for delta temp:

```
public static ContinuousSet risingTemperature;
public static ContinuousSet fallingTemperature;
public static ContinuousSet constantTemperature;
```

//for delta pressure:

```
public static ContinuousSet weatherGettingBetter;
public static ContinuousSet weatherNoChange;
public static ContinuousSet weatherGettingWorse;
```

//ASTRONOMY

```

//moon & sun peak times:
public static ContinuousSet moonRise;
public static ContinuousSet moonSet;
public static ContinuousSet sunRise;
public static ContinuousSet sunSet;

//CONSEQUENT
//for action
public static ContinuousSet lowActivity;
public static ContinuousSet mediumActivity;
public static ContinuousSet highActivity;

```

Инициализиране

```

protected void buildActivitySets()
{
    fishActivity = new ContinuousDimension("FishActivity", "Low to High",
    "", -10, 10);

    lowActivity = new RightQuadraticSet(fishActivity, "Low activity", -10,
-5, 5);
    highActivity = new LeftQuadraticSet(fishActivity, "High activity", -5,
5, 10);
    mediumActivity = new BellSet(fishActivity, "Medium activity", 0, 3, 5);
}

private void BuildMoonSunPeakSets(string sunrise, string sunset, string
moonrise, string moonset)
{
    var sunriseTime = DateHelper.ParseDate(sunrise).TimeOfDay;
    var sunsetTime = DateHelper.ParseDate(sunset).TimeOfDay;
    var moonriseTime = DateHelper.ParseDate(moonrise).TimeOfDay;
    var moonsetTime = DateHelper.ParseDate(moonset).TimeOfDay;

    BuildMoonSunPeakSets(sunriseTime, sunsetTime, moonriseTime,
moonsetTime);
}

private void BuildMoonSunPeakSets(TimeSpan sunrise, TimeSpan sunset,
TimeSpan moonrise, TimeSpan moonset)
{
    dayTime = new ContinuousDimension("time", "Time of the day", "t,m", -
120, 24 * 60 - 1);

    int sunriseTime = (int)sunrise.TotalMinutes;
    sunRise = new BellSet(dayTime, "Sunrise", sunriseTime, 45, 120);

    int sunsetTime = (int)sunset.TotalMinutes;
    sunSet = new BellSet(dayTime, "Sunset", sunsetTime, 45, 120);

    int moonriseTime = (int)moonrise.TotalMinutes;
    moonRise = new BellSet(dayTime, "Moonrise", moonriseTime, 45, 120);

    int moonsetTime = (int)moonset.TotalMinutes;
    moonSet = new BellSet(dayTime, "Moonset", moonsetTime, 45, 120);
}

private void BuildTemperatureSets(int minTemp, int avgTemp, int maxTemp)

```

```

{
    temperature = new ContinuousDimension("t", "Temperature - weather",
"C", minTemp, maxTemp);

    lowTemperature = new RightQuadraticSet(temperature, "Low temperature",
avgTemp - 10, avgTemp - 5, avgTemp);
    highTemperature = new LeftQuadraticSet(temperature, "High temperature",
avgTemp, avgTemp + 5, avgTemp + 10);
    goodTemperature = new BellSet(temperature, "Good temperature", avgTemp,
5, 10);
}

private void BuildDeltaTemperatureSets()
{
    deltaTemperature = new ContinuousDimension("?t", "Change of temperature
for past minutes", "C/t", -10, +10);

    fallingTemperature = new RightQuadraticSet(deltaTemperature, "Falling
temperature", -6, -2, 0);
    risingTemperature = new LeftQuadraticSet(deltaTemperature, "Rising
temperature", 0, 2, 6);
    constantTemperature = new BellSet(deltaTemperature, "Constant
temperature", 0, 2, 5);
}

private void BuildDeltaPressureSets()
{
    int normalPressure = 0;//normal
    int changeStep = 100;

    deltaPressure = new ContinuousDimension("?t", "Change of temperature
for past minutes", "C/t", normalPressure - changeStep * 2, normalPressure +
changeStep * 2);

    weatherGettingBetter = new RightQuadraticSet(deltaPressure, "Getting
better (Falling pressure)", normalPressure - changeStep, normalPressure -
changeStep / 2, normalPressure);
    weatherGettingWorse = new LeftQuadraticSet(deltaPressure, "Getting
worse (Rising pressure)", normalPressure, normalPressure + changeStep / 2,
normalPressure + changeStep);
    weatherNoChange = new BellSet(deltaPressure, "Constant temperature",
normalPressure, changeStep / 2, changeStep);
}

```

Изпълнение на изчисленията между зависимостите

//Метод за извършване на изчисленията

```
protected void buildRelationNow(bool initial)
```

```

{
    if (!_ready)
return;

    _waitingForBuild = false;
    _building = true;

    bool _expressionChanged = false;

    //Вземаме входящите данни
    decimal inputTemperature = txtTemp.Value;
    decimal inputDeltaTemperature = txtDeltaTemp.Value;
    int inputDayTime = (int)dayTimePicker.Value.TimeOfDay.TotalMinutes;

```

```

        decimal inputDeltaPressurre = txtDeltaPressure.Value;

        //Оценяваме релацията от потребителския интерфейс, като го компилираме
до C# код
        #region Expression evaluation with C#
        string strExpression = txtExpression.Text;
        PrependFullName(ref strExpression, "lowTemperature");
        PrependFullName(ref strExpression, "highTemperature");
        PrependFullName(ref strExpression, "goodTemperature");
        PrependFullName(ref strExpression, "risingTemperature");
        PrependFullName(ref strExpression, "fallingTemperature");
        PrependFullName(ref strExpression, "constantTemperature");
        PrependFullName(ref strExpression, "weatherGettingBetter");
        PrependFullName(ref strExpression, "weatherNoChange");
        PrependFullName(ref strExpression, "weatherGettingWorse");
        PrependFullName(ref strExpression, "moonRise");
        PrependFullName(ref strExpression, "moonSet");
        PrependFullName(ref strExpression, "sunRise");
        PrependFullName(ref strExpression, "sunSet");
        PrependFullName(ref strExpression, "lowActivity");
        PrependFullName(ref strExpression, "highActivity");
        PrependFullName(ref strExpression, "mediumActivity");

        object obj = Evaluator.Eval(strExpression);

        if (obj != null)
        {
        if (!(obj is FuzzyRelation))
        {
            MessageBox.Show(String.Format("ERROR: Object of type FuzzyRelation
expected as the result of the expression.\r\nThis object is type {0}.",
obj.GetType().FullName),
            "Error evaluating expression", MessageBoxButtons.OK,
            MessageBoxIcon.Error);
        }
        else
        {
            //Вземаме резултата от оценяването
            _relation = (FuzzyRelation)obj;
            if (_expression != txtExpression.Text)
            _expressionChanged = true;
            _expression = txtExpression.Text;
        }
        }
        #endregion

        //Дефъзификация, чрез използване на Center of Gravity
        #region Defuzzification
        DefuzzificationFactory.DefuzzificationMethod method =
        DefuzzificationFactory.DefuzzificationMethod.CenterOfGravity;
        _defuzzification = DefuzzificationFactory.GetDefuzzification(
        _relation,
        new Dictionary<IDimension, decimal> {
        { temperature, inputTemperature },
        { deltaTemperature, inputDeltaTemperature },
        { dayTime, inputDayTime },
        { deltaPressure, inputDeltaPressurre }
        },
        method
        );

```

```
_defuzzMethod = method;
#endregion

#region Output value
string unit =
((IContinuousDimension)_defuzzification.OutputDimension).Unit;
lblOutput.Text = _defuzzification.CrispValue.ToString("F5") +
(string.IsNullOrEmpty(unit) ? "" : " " + unit);
#endregion

_building = false;
}
```