Design and Implementation of Neutrosophic Data Operations Using Object Oriented Programming

A. A. Salama[1], , Mohamed Abdelfattah[2,]H. A. El-Ghareeb[3,]A. M. Manie[4]
[1,4]Department of Mathematics and Computer Science, Faculty of Sciences, Port Said University, Egypt.
[1*]Drsalama44@gmail.com
[2]Information System Department, Faculty of Computers & Information, Benha University, EGYPT
[2*]Information System Department, Faculty of Computers & Information, Islamic University, KSA
[3] Department of Information Systems, Faculty of Computers and Information Sciences, Mansoura University, Egypt

Abstract
It is of great interest to develop a framework containing a library of the main building blocks of common use to neutrosophic data. This framework would save much of the routine work that is needed to build most types of neutrosophic data. In this paper, we would like to introduce model neutrosophic data sets. The use of object oriented programming techniques and concepts as they may apply to the design and development a new framework to implement neutrosophic data operations

Keywords: Neutrosophicdata, software programs, neutrosophic database. AMS Subject Classification: 03B99, 03E99

1 Introduction
In real-life problems, the data associated are often imprecise, or non-deterministic. All real data cannot be precise because of their fuzzy nature. Imprecision can be of many types: non-matching data values, imprecise queries, inconsistent data misaligned schemas, etc. The fundamental concepts of neutrosophic set, introduced by Smarandache in [8, 9] and Salamaet al. in [1, 2, 3, 4, 5, 6 7, 10, 11, 12, 13, 15, 16], provides a natural foundation for treating mathematically the neutrosophic phenomena which exist pervasively in our real world and for building new branches of neutrosophic mathematics.
In this paper, we have developed an Excel package to be utilized for calculating neutrosophic data and analyze them. We have used Excel as it is a powerful tool that is widely accepted and used for statistical analysis. In this paper, we have developed an Excel package to be utilized for calculating neutrosophic data and analyze them. The use of object oriented programming techniques and concepts as they may apply to the design and development a new framework to implement neutrosophic data operations, the c# programming language, NET Framework and Microsoft Visual Studio are used to implement the neutrosophic classes.

2 Related Works
We recollect some relevant basic preliminaries, and in particular, the work of Smarandache in [8, 9], and Salamaet al. [1, 2, 3, 4, 5, 6, 7, 10, 11, 12, 13, 15, 16]. The c# programming

language, NET Framework and Microsoft Visual Studio are used to implement the neutrosophic classes.

3 A Proposed frameworks

Fig. 1 shows Class Diagram of the implemented package. Fig. 2 presents a working example of the package interface calculating the complement. Our implemented neutrosophic package can calculate Intersection, Union, and Complement of the nuetrosophic set. Fig. 3 presents our neutrosphic package capability to draw figures of presented neutrosphic set. Fig. 4 presents charting of Union operation calculation, and Fig. 5 Intersection operation nuetrosophic set are characterized by its efficiency as it takes into consideration the three data items: True, Indeterminate and False We introduce the neutrosophic package class diagram.

4 Neutrosophic Operations Codes

Object oriented programming languages offer benefits toward the approach and design of software programs. The Primary advantages include code reusability, rapid prototyping, as well as, lower software design and maintenance costs. The use of object oriented programming techniques and concepts as they may apply to the design and development a new framework to implement Neutrosophic operations. The c# programming language, .NET Framework and Microsoft Visual Studio is used to implement.In the following, some c# code attached are used to explain the concept described inabove Section more clearly.theNeutrosophicclassess fuzzy operations can copy the operand fuzzy set'smembership function class to create resultant fuzzy set.

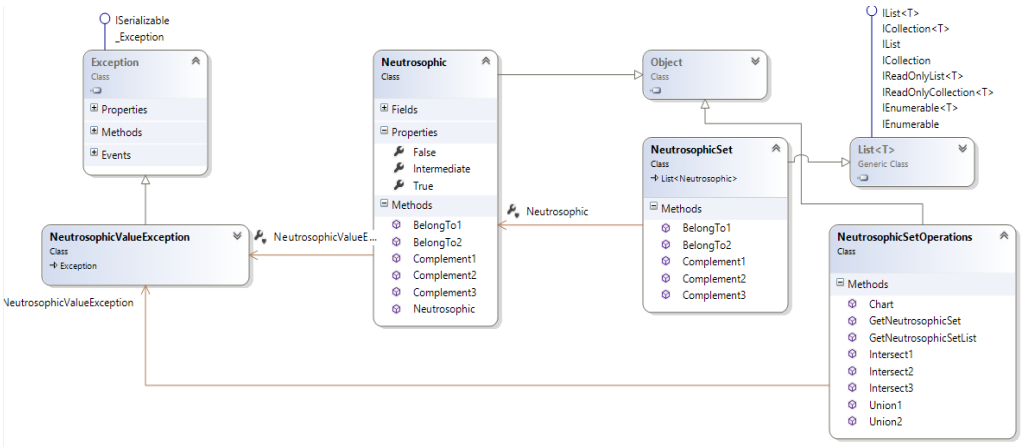
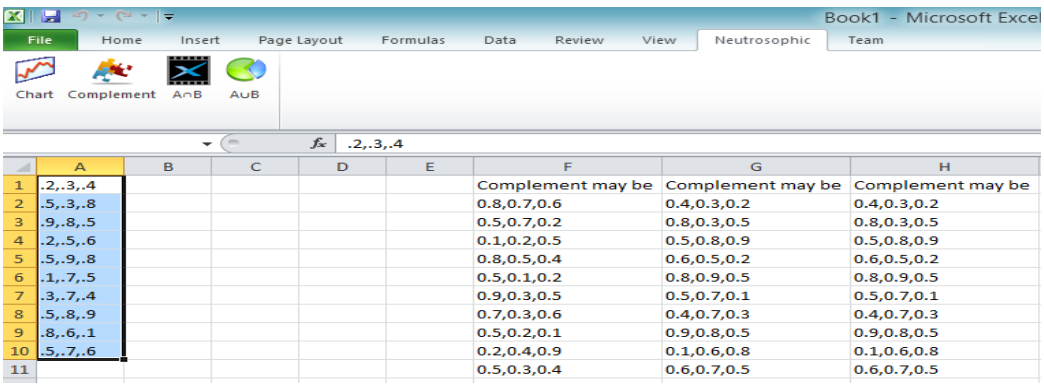


Fig. 1. Neutrosophic package class diagram



	A	B	C	D	E	F	G	H
1	.2,.3,.4					Complement may be	Complement may be	Complement may be
2	.5,.3,.8					0.8,0.7,0.6	0.4,0.3,0.2	0.4,0.3,0.2
3	.9,.8,.5					0.5,0.7,0.2	0.8,0.3,0.5	0.8,0.3,0.5
4	.2,.5,.6					0.1,0.2,0.5	0.5,0.8,0.9	0.5,0.8,0.9
5	.5,.9,.8					0.8,0.5,0.4	0.6,0.5,0.2	0.6,0.5,0.2
6	.1,.7,.5					0.5,0.1,0.2	0.8,0.9,0.5	0.8,0.9,0.5
7	.3,.7,.4					0.9,0.3,0.5	0.5,0.7,0.1	0.5,0.7,0.1
8	.5,.8,.9					0.7,0.3,0.6	0.4,0.7,0.3	0.4,0.7,0.3
9	.8,.6,.1					0.5,0.2,0.1	0.9,0.8,0.5	0.9,0.8,0.5
10	.5,.7,.6					0.2,0.4,0.9	0.1,0.6,0.8	0.1,0.6,0.8
11						0.5,0.3,0.4	0.6,0.7,0.5	0.6,0.7,0.5

Fig. 2. Neutrosophic package interface and calculating complement

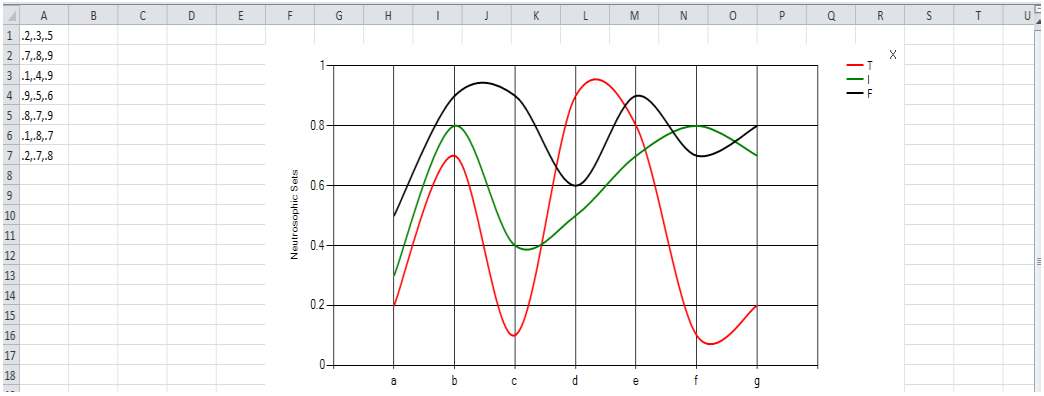


Fig. 3.Neutrosophicchart

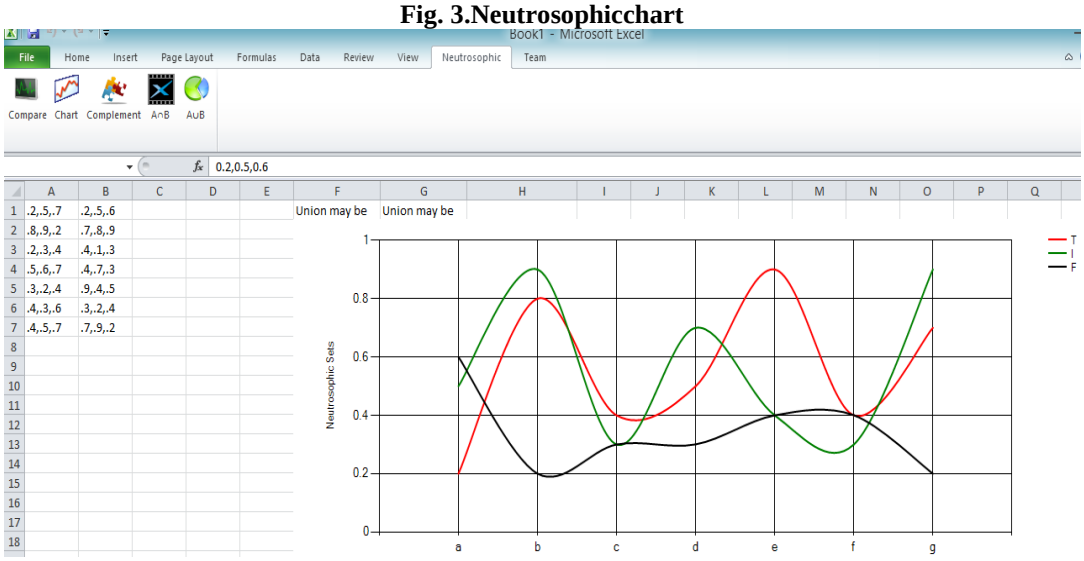


Fig. 4.Neutrosophicpackage union chart

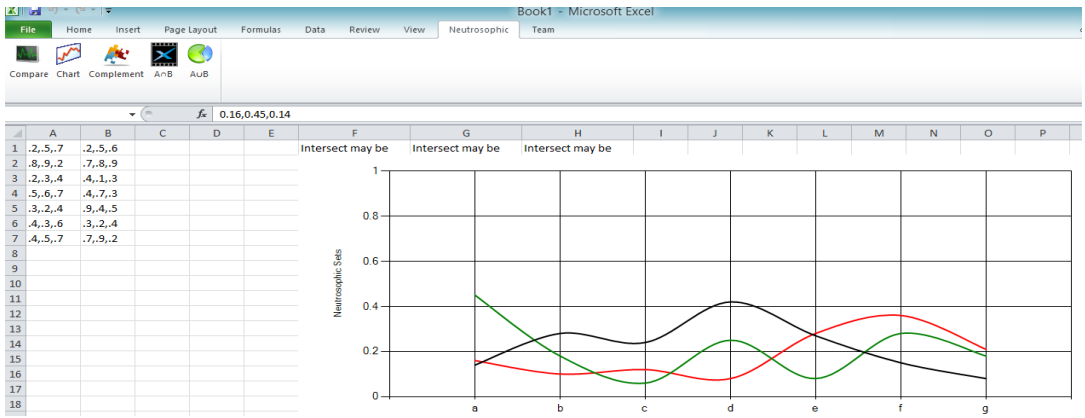


Fig. 5. Neutrosophic package intersection chart

```
usingSystem.Collections.Generic;
usingSystem.Linq;
usingSystem.Text;
namespaceRibbonCustomize
{
classNeutrosophicValueException:Exception
   {
publicNeutrosophicValueException()
```

```
        : base("Neutrosophic value must be between 0 and 1")
      {
      }
   }
classNeutrosophicSet:List<Neutrosophic>
   {
publicNeutrosophicSet Complement1()
      {
NeutrosophicSetcomplementSet = newNeutrosophicSet();
foreach (Neutrosophic n inthis)
         {
complementSet.Add(n.Complement1());
         }
returncomplementSet;
      }
publicNeutrosophicSet Complement2()
      {
NeutrosophicSetcomplementSet = newNeutrosophicSet();
foreach (Neutrosophic n inthis)
         {
complementSet.Add(n.Complement2());
         }
returncomplementSet;
      }
publicNeutrosophicSet Complement3()
      {
NeutrosophicSetcomplementSet = newNeutrosophicSet();
foreach (Neutrosophic n inthis)
         {
complementSet.Add(n.Complement3());
         }
returncomplementSet;
      }
publicBoolean

BelongTo1(NeutrosophicSetnSet)
      {
for (inti = 0; i<this.Count; i++)
         {
if (!this[i].BelongTo1(nSet[i]))
returnfalse;
         }
returntrue;
      }
publicBoolean BelongTo2(NeutrosophicSetnSet)
      {
for (inti = 0; i<this.Count; i++)
         {
if (!this[i].BelongTo2(nSet[i]))
returnfalse;
         }
returntrue;
      }

   }

classNeutrosophic
   {
double t, i, f;
```

```
publicNeutrosophic(doublet,doublei,double f)
      {
         T = t;
         I = i;
         F = f;
      }
publicdouble T
      {
get
         {
returnConvert.ToDouble( Math.Round( t,4));
         }
set
         {
if (t < 0 || t > 1)
thrownewNeutrosophicValueException();
            t = value;
         }
      }
publicdouble I
      {
get
         {
returnConvert.ToDouble(Math.Round(i, 4));
         }
set
         {
if (value< 0 || value> 1)
thrownewNeutrosophicValueException();
i = value;
         }
      }
publicdouble F
      {
get
         {
returnConvert.ToDouble(Math.Round(f, 4)); ;
         }
set
         {
if (value< 0 || value> 1)
thrownewNeutrosophicValueException();
            f = value;
         }
      }
publicNeutrosophic Complement1()
      {
Neutrosophic complement = newNeutrosophic(1 - T, 1 - i, 1 - F);
return complement;
      }
publicNeutrosophic Complement2()
      {
Neutrosophic complement = newNeutrosophic(F,i, T);
return complement;
      }
publicNeutrosophic Complement3()
      {
Neutrosophic complement = newNeutrosophic(F, i, T);
return complement;
      }
```

```
publicbool BelongTo1(Neutrosophic n)
        {
return (this.T<= n.T&&this.I<= n.I&&this.F>= n.F);
        }
publicbool BelongTo2(Neutrosophic n)
        {
return (this.T<= n.T&&this.I>= n.I&&this.F>= n.F);
        }
    }
}
using System;
usingSystem.Collections.Generic;
usingSystem.Linq;
usingSystem.Text;
usingSystem.Windows.Forms.DataVisualization.Charting;
usingSystem.Drawing;
using Excel = Microsoft.Office.Interop.Excel;
namespaceRibbonCustomize
{
classNeutrosophicSetOperations
    {
publicstaticvoid Chart(ChartmyChart, paramsNeutrosophicSet[] nsArray)
        {
intnsOrder = 1;
intnCount = nsArray[0].Count;
intlabelIndex = 0;
foreach (NeutrosophicSet ns innsArray)
          {
string T = "T";
string I = "I";
string F = "F";
if (nsArray.Length> 1)
              {
                 T = "T" + nsOrder;
                 I = "I" + nsOrder;
                 F = "F" + nsOrder;
              }
myChart.Series.Add(T);
myChart.Series.Add(I);
myChart.Series.Add(F);
myChart.Series[T].Color = Color.Red;
myChart.Series[I].Color = Color.Green;
myChart.Series[F].Color = Color.Blue;

intyValue = 97;
foreach (Neutrosophic n in ns)
              {
string c = Convert.ToChar(yValue).ToString();
myChart.Series[T].Points.AddXY(c, n.T);
myChart.Series[I].Points.AddXY(c, n.I);
myChart.Series[F].Points.AddXY(c, n.F);
yValue++;
              }
FontFamily family = newFontFamily("Times New Roman");
Fontfont = newFont(family, 12, FontStyle.Bold | FontStyle.Italic);
myChart.Series[T].Points[0].Label = T;
myChart.Series[T].Points[0].Font = font;
myChart.Series[I].Points[0].Label = I;
myChart.Series[I].Points[0].Font = font;
myChart.Series[F].Points[0].Label = F;
```

```
myChart.Series[F].Points[0].Font = font;
nsOrder++;
        }
foreach (SeriesserinmyChart.Series)
        {
ser.BorderWidth = 2;
ser.ChartType = SeriesChartType.Spline;
        }
    }
publicstaticNeutrosophicSet Intersect1(paramsNeutrosophicSet[] nSetArray)
    {
intneutroSophicCount = nSetArray[0].Count;
NeutrosophicSetintersectSet = newNeutrosophicSet();
for (int x = 0; x <neutroSophicCount; x++)
        {
double t = 1, i = 1, f = 1;
for (int y = 0; y <nSetArray.Length; y++)
            {
                t *= nSetArray[y][x].T;
i *= nSetArray[y][x].I;
                f *= nSetArray[y][x].F;
            }
Neutrosophicneutro = newNeutrosophic(t, i, f);
intersectSet.Add(neutro);
        }
returnintersectSet;
    }
publicstaticNeutrosophicSet Intersect2(paramsNeutrosophicSet[] nSetArray)
    {
intneutroSophicCount = nSetArray[0].Count;
NeutrosophicSetintersectSet = newNeutrosophicSet();
for (int x = 0; x <neutroSophicCount; x++)
        {
doubleminT = nSetArray[0][x].T;
doubleminI = nSetArray[0][x].I;
doublemaxF = nSetArray[0][x].F;
for (int y = 1; y <nSetArray.Length; y++)
            {
if (nSetArray[y][x].T <minT)
minT = nSetArray[y][x].T;
if (nSetArray[y][x].I <minI)
minI = nSetArray[y][x].I;
if (nSetArray[y][x].F >maxF)
maxF = nSetArray[y][x].F;
            }
intersectSet.Add(newNeutrosophic(minT, minI, maxF));
        }
returnintersectSet;
    }
publicstaticNeutrosophicSet Intersect3(paramsNeutrosophicSet[] nSetArray)
    {
intneutroSophicCount = nSetArray[0].Count;
NeutrosophicSetintersectSet = newNeutrosophicSet();
for (int x = 0; x <neutroSophicCount; x++)
        {
doubleminT = nSetArray[0][x].T;
doublemaxI = nSetArray[0][x].I;
doublemaxF = nSetArray[0][x].F;
for (int y = 1; y <nSetArray.Length; y++)
            {
```

```
if (nSetArray[y][x].T <minT)
minT = nSetArray[y][x].T;
if (nSetArray[y][x].I >maxI)
maxI = nSetArray[y][x].I;
if (nSetArray[y][x].F >maxF)
maxF = nSetArray[y][x].F;
            }
intersectSet.Add(newNeutrosophic(minT, maxI, maxF));
        }
returnintersectSet;
    }
publicstaticNeutrosophicSet Union1(paramsNeutrosophicSet[] nSetArray)
    {
intneutroSophicCount = nSetArray[0].Count;
NeutrosophicSetintersectSet = newNeutrosophicSet();

for (int x = 0; x <neutroSophicCount; x++)
        {
doublemaxT = nSetArray[0][x].T;
doublemaxI = nSetArray[0][x].I;
doubleminF = nSetArray[0][x].F;
for (int y = 1; y <nSetArray.Length; y++)
            {
if (nSetArray[y][x].T >maxT)
maxT = nSetArray[y][x].T;
if (nSetArray[y][x].I >maxI)
maxI = nSetArray[y][x].I;
if (nSetArray[y][x].F <minF)
minF = nSetArray[y][x].F;
            }
intersectSet.Add(newNeutrosophic(maxT, maxI, minF));
        }
returnintersectSet;
    }
publicstaticNeutrosophicSet Union2(paramsNeutrosophicSet[] nSetArray)
    {
intneutroSophicCount = nSetArray[0].Count;
NeutrosophicSetintersectSet = newNeutrosophicSet();
for (int x = 0; x <neutroSophicCount; x++)
        {
doublemaxT = nSetArray[0][x].T;
doubleminI = nSetArray[0][x].I;
doubleminF = nSetArray[0][x].F;
for (int y = 1; y <nSetArray.Length; y++)
            {
if (nSetArray[y][x].T >maxT)
maxT = nSetArray[y][x].T;
if (nSetArray[y][x].I <minI)
minI = nSetArray[y][x].I;
if (nSetArray[y][x].F <minF)
minF = nSetArray[y][x].F;
            }
intersectSet.Add(newNeutrosophic(maxT, minI, minF));
        }
returnintersectSet;
    }
publicstaticNeutrosophicSetGetNeutrosophicSet(Excel.Range range)
    {
NeutrosophicSetnSet = newNeutrosophicSet();
string[] arrVal = newstring[range.Count];
```

```
int index = 0;
foreach (Excel.Range r inrange.Cells)
        {
arrVal[index++] = (Convert.ToString(r.Value2));
        }
for (int r = 0; r <arrVal.Length; r++)
        {
string[] strArr = arrVal[r].Split(',');
nSet.Add(newNeutrosophic(double.Parse(strArr[0]), double.Parse(strArr[1]), double.Parse(strArr[2])));
        }
returnnSet;
    }
publicstaticList<NeutrosophicSet>GetNeutrosophicSetList(Excel.Range range)
    {
List<NeutrosophicSet>nSetList = newList<NeutrosophicSet>();
boolstartFlag = true, againFlag = false;
charstartCol = ' ', currentCol = ' ', prevCol = ' ';
intnSetIndex = 0;
foreach (Excel.Range r inrange.Cells)
        {
if (startFlag)
            {
startCol = r.Address.ToString()[1];
startFlag = false;
nSetList.Add(newNeutrosophicSet());
prevCol = startCol;
            }
else
            {
currentCol = r.Address.ToString()[1];
if (currentCol == startCol)
                {
againFlag = true;
nSetIndex = -1;
                }
if (againFlag)
nSetIndex++;
elseif (currentCol != prevCol)
                {
prevCol = currentCol;
nSetList.Add(newNeutrosophicSet());
nSetIndex++;
                }
            }
stringcellValue = (Convert.ToString(r.Value2));
string[] strArr = cellValue.Split(',');
Neutrosophic n = newNeutrosophic(double.Parse(strArr[0]), double.Parse(strArr[1]), double.Parse(strArr[2]));
nSetList[nSetIndex].Add(n);
        }
returnnSetList;
    }
  }
}
using System;
usingSystem.Collections.Generic;
usingSystem.Linq;
usingSystem.Text;
usingMicrosoft.Office.Tools.Ribbon;
using Excel = Microsoft.Office.Interop.Excel;
usingSystem.Windows.Forms;
```

```
usingSystem.Collections;
usingMicrosoft.Office.Tools.Excel;
usingFormChart = System.Windows.Forms.DataVisualization.Charting.Chart;
usingSystem.Windows.Forms.DataVisualization.Charting;
usingSystem.Drawing;
namespaceRibbonCustomize
{
publicpartialclassRibbon1
   {
privatevoid Ribbon1_Load(object sender, RibbonUIEventArgs e)
      {
      }
FormChartmyChart;
privatevoidbtnChart_Click(object sender, RibbonControlEventArgs e)
      {
Excel.Rangeselectedrange = Globals.ThisAddIn.Application.Selection;
myChart = newFormChart();
Worksheetworksheet                                                                                    =
Globals.Factory.GetVstoObject(Globals.ThisAddIn.Application.ActiveWorkbook.ActiveSheet);
Excel.Range cells = worksheet.Range["F2", "R20"];
worksheet.Controls.AddControl(myChart, cells, "MyChart");
//myChart.Legends.Add(new Legend("LG"));
myChart.ChartAreas.Add("area");
myChart.ChartAreas["area"].AxisY.Minimum = 0;
myChart.ChartAreas["area"].AxisY.Maximum = 1;
myChart.ChartAreas["area"].AxisY.Title = "Neutrosophic Set";
if (selectedrange.Columns.Count == 1)
         {
NeutrosophicSetnSet = NeutrosophicSetOperations.GetNeutrosophicSet(selectedrange);
NeutrosophicSetOperations.Chart(myChart, nSet);
         }
else
         {
List<NeutrosophicSet>nSetList = NeutrosophicSetOperations.GetNeutrosophicSetList(selectedrange);
NeutrosophicSetOperations.Chart(myChart, nSetList.ToArray());
         }
Buttonbtn = newButton();
btn.Text = "X";
btn.Width = 30;
btn.FlatStyle = FlatStyle.Flat;
btn.FlatAppearance.BorderSize = 0;
btn.Location = newPoint(myChart.Width - btn.Width, 0);
myChart.Controls.Add(btn);
btn.Click += btn_Click;

Labellbl = newLabel();
lbl.Text = "Chart type";
lbl.TextAlign = ContentAlignment.MiddleRight;
lbl.Location = newPoint(10, myChart.Height - lbl.Height - 5);
ComboBoxcmb = newComboBox();
cmb.DropDownStyle = ComboBoxStyle.DropDownList;
cmb.Location = newPoint(lbl.Location.X + lbl.Width + 2, myChart.Height - lbl.Height - 5);
cmb.SelectedIndexChanged += cmb_SelectedIndexChanged;
foreach (string value inEnum.GetNames(typeof(SeriesChartType)))
         {
cmb.Items.Add(value);
         }
myChart.Controls.Add(lbl);
myChart.Controls.Add(cmb);
      }
```

```
voidbtn_Click(object sender, EventArgs e)
        {
Worksheetworksheet =
Globals.Factory.GetVstoObject(Globals.ThisAddIn.Application.ActiveWorkbook.ActiveSheet);
worksheet.Controls.RemoveAt(0);
        }
voidcmb_SelectedIndexChanged(object sender, EventArgs e)
        {
ComboBoxcmb = (ComboBox)sender;
SeriesChartType type = (SeriesChartType)Enum.Parse(typeof(SeriesChartType), cmb.SelectedItem.ToString());
foreach (SeriesserinmyChart.Series)
ser.ChartType = type;
        }
privatevoidbtnComplement_Click(object sender, RibbonControlEventArgs e)
        {
Excel.Rangeselectedrange = Globals.ThisAddIn.Application.Selection;
NeutrosophicSetnSet = NeutrosophicSetOperations.GetNeutrosophicSet(selectedrange);
NeutrosophicSet nSetcom1 = nSet.Complement1();
NeutrosophicSet nSetcom2 = nSet.Complement2();
NeutrosophicSet nSetcom3 = nSet.Complement3();
PrintResult("F", "Complement may be", 19, nSetcom1);
PrintResult("G", "Complement may be", 19, nSetcom2);
PrintResult("H", "Complement may be", 19, nSetcom3);
        }
voidPrintResult(stringcolumnName, stringmsg, intcolumnWidth, NeutrosophicSetnSet)
        {
WorksheetactiveWorksheet =
Globals.Factory.GetVstoObject(Globals.ThisAddIn.Application.ActiveWorkbook.ActiveSheet);
Excel.Range rangeCom1 = activeWorksheet.get_Range(columnName + "1");
          rangeCom1.Value2 = msg;
rangeCom1.EntireColumn.ColumnWidth = columnWidth;
int f = 2;
foreach (Neutrosophic n innSet)
          {
Excel.RangerangeComValue = activeWorksheet.get_Range(columnName + (f++));
              rangeComValue.Value2 = "";
              rangeComValue.Value2 += "," + n.T;
              rangeComValue.Value2 += "," + n.I;
              rangeComValue.Value2 += "," + n.F;
              rangeComValue.Value2 = ((string)rangeComValue.Value2).Remove(0, 1);
          }
//for (int r = 0; r <arrayToPrint.GetLength(0); r++)
//{
//    Excel.RangerangeComValue = activeWorksheet.get_Range(columnName + (f++));

//    rangeComValue.Value2 = "";
//    for (int c = 0; c <arrayToPrint.GetLength(1); c++)
//    {
//        rangeComValue.Value2 += "," + arrayToPrint[r, c];
//    }
//    rangeComValue.Value2 = ((string)rangeComValue.Value2).Remove(0, 1);
//}
        }
privatevoidbtnIntersect_Click(object sender, RibbonControlEventArgs e)
        {
WorksheetactiveWorksheet =
Globals.Factory.GetVstoObject(Globals.ThisAddIn.Application.ActiveWorkbook.ActiveSheet);
activeWorksheet.Range["F1", "H26"].Clear();
Excel.Range range = Globals.ThisAddIn.Application.Selection;
List<NeutrosophicSet>nSetList = NeutrosophicSetOperations.GetNeutrosophicSetList(range);
```

```
NeutrosophicSet set1 = NeutrosophicSetOperations.Intersect1(nSetList.ToArray());
NeutrosophicSet set2 = NeutrosophicSetOperations.Intersect2(nSetList.ToArray());
NeutrosophicSet set3 = NeutrosophicSetOperations.Intersect3(nSetList.ToArray());
PrintResult("F", "Intersect may be", 18, set1);
PrintResult("G", "Intersect may be", 18, set2);
PrintResult("H", "Intersect may be", 18, set3);
      }
privatevoidbtnUnion_Click(object sender, RibbonControlEventArgs e)
      {
Worksheetactive Worksheet =
Globals.Factory.GetVstoObject(Globals.ThisAddIn.Application.ActiveWorkbook.ActiveSheet);
activeWorksheet.Range["F1", "H26"].Clear();
Excel.Range range = Globals.ThisAddIn.Application.Selection;
List<NeutrosophicSet>nSetList = NeutrosophicSetOperations.GetNeutrosophicSetList(range);
NeutrosophicSet set1 = NeutrosophicSetOperations.Union1(nSetList.ToArray());
NeutrosophicSet set2 = NeutrosophicSetOperations.Union2(nSetList.ToArray());
PrintResult("F", "Intersect may be", 18, set1);
PrintResult("G", "Intersect may be", 18, set2);
      }
   }
}
```

5 Conclusions and Future Work

The advantage to using object-oriented modelling of Neutrosophics sets is that we avoid problems of the distortion problem in general. Furthermore, because the result of Neutrosophic set contains all the Neutrosophic membership functions and operators involved, the Neutrosophic set operations are traceable. In addition, due to our object-oriented structure, the Neutrosophicmembership functions and operators can be reused and managed in a flexible way. Although increasing cascaded operations will increase the number of computation, this overhead is not large. On the other hand, grades of membership will only be calculated when needed and avoid unnecessary computation of traditional approaches. Therefore in general the performance of our approach will be better. The traditional method of modelling neutrosophic data sets with a pair of arrays to store elements and their grades of membership may cause distortion if sampling is not enough. Our object-oriented model solves this problem. Our program not only solves the distortion problem, but also provides a flexible and efficient way to develop full-scaled fuzzy logic systems.In future studies we will develop some software programs to deal with the statistical analysis of the neutrosophic data in [1-16].

CompetingInterests

Authors have declared that no competing interests exist.

Acknowledgement

I gratefully acknowledge the support and generosity of Benha university, faculty of computers and informatics http://www.bu.edu.eg/

References

[1] S. A. Alblowi, A. A. Salama&MohmedEisa, New Concepts of Neutrosophic Sets, International Journal of Mathematics and Computer Applications Research (IJMCAR), Vol. 4, Issue 1, 59-66, 2014

[2] I. M. Hanafy, A.A. Salama and K. Mahfouz, Correlation of neutrosophic Data, International Refereed Journal of Engineering and Science (IRJES) , Vol.(1), Issue 2.(2012) PP.39-43

[3] I.M. Hanafy, A.A. Salama and K.M. Mahfouz,," Neutrosophic Classical Events and Its Probability" International Journal of Mathematics and Computer Applications Research(IJMCAR) Vol.(3),Issue 1,Mar (2013), pp171-178.

[4] A.A. Salama and S.A. Alblowi, "Generalized Neutrosophic Set and Generalized Neutrousophic Topological Spaces ", Journal computer Sci. Engineering, Vol. (2) No. (7) (2012)pp129-132.

[5] A.A.Salama and S.A.Alblowi, Neutrosophic set and neutrosophic topological space, ISORJ. Mathematics,Vol.(3), Issue(4), .(2012) pp-31-35.

[6] A.A. Salama and S.A.Alblowi, Intuitionistic Fuzzy Ideals Topological Spaces, Advances in Fuzzy Mathematics , Vol.(7), Number 1, (2012).pp. 51- 60

[7] A.A.Salama, and H.Elagamy, "Neutrosophic Filters" International Journal of Computer Science Engineering and Information Technology Reseearch (IJCSEITR), Vol.3, Issue(1),Mar 2013, .(2013), pp 307-312

[8] FlorentinSmarandache, Neutrosophy and Neutrosophic Logic , First International Conference on Neutrosophy , NeutrosophicLogic , Set, Probability, and Statistics University of New Mexico, Gallup, NM 87301, USA(2002).

[9] F.Smarandache. A Unifying Field in Logics: Neutrosophic Logic. Neutrosophy, Neutrosophic Set, Neutrosophic Probability, American Research Press, Rehoboth, NM, (1999).

[10] A. A. Salama, "Neutrosophic Crisp Points &Neutrosophic Crisp Ideals", Neutrosophic Sets and Systems, Vol.1, No. 1,(2013) pp 50-54.

[11] A. A. Salama and F. Smarandache, "Filters via Neutrosophic Crisp Sets", Neutrosophic Sets and Systems, Vol.1, No. 1,(2013) pp34-38.

[12] A. A. Salama, F. Smarandache and ValeriKroumov, "Neutrosophic Crisp Sets &Neutrosophic Crisp Topological Spaces", Neutrosophic Sets and Systems , Vol.(2) pp25-30, 2014.

[13] A. A. Salama and F. Smarandache and S. A. Alblowi "The Characteristic Function of a Neutrosophic Set "Neutrosophic Sets and Systems,(2014) (Accepted).

[14] A.A.Salama," The Concept of Neutrosophic Set and Basic Properties of Neutrosophic Set Operations" WASET 2012 PARIS, FRANC, International University of Science, Engineering and Technology,2012.

[15] A. A. Salama, Mohamed Eisa and M. M . Abdelmoghny, "Neutrosophic Relations Database" International Journal of Information Science and Intelligent System, 3(1) (2014).

[16] A. A. Salama, Said Broumi and FlorentinSmarandache, Neutrosophic Crisp Open Set and Neutrosophic Crisp Continuity via Neutrosophic Crisp Ideals, I.J. Information Engineering and Electronic Business, 2014, Published Online October 2014 in MECS.