Raymond BISDORFF

Algorithmic Decision Making with Python Resources

Tutorials and Advanced Topics

July 6, 2021

Springer

Contents

Chapter 1
Working with outranking digraphs

Abstract To be written.

1.1 Outranking digraph model

In the `outrankingDigraphs` module, the `BipolarOutrankingDigraph` class provides our standard **outranking digraph** constructor. Such an instance represents a **hybrid** object of both, the `PerformanceTableau` type and the `OutrankingDigraph` type. A given object contains hence the following attributes:

1. A ordered dictionary of decision **actions** describing the potential decision actions or alternatives with `name` and `comment` attributes,
2. A possibly empty ordered dictionary of decision **objectives** with `name` and `comment` attributes, describing the multiple preference dimensions involved in the decision problem,
3. An ordered dictionary of performance **criteria**, i.e. *preferentially independent* and *non-redundant* decimal-valued functions used for measuring the performance of each potential decision action with respect to a decision objective,
4. A double dictionary called **evaluation** gathering performance grades for each decision action or alternative on each criterion function.
5. The digraph **valuationdomain**, a dictionary with three entries: the *minimum* (-1.0, certainly outranked), the *median* (0.0, indeterminate) and the *maximum* characteristic value ($+1.0$, certainly outranking),
6. The outranking **relation** : a double dictionary defined on the Cartesian product of the set of decision alternatives capturing the credibility of the pairwise *outranking situation* computed on the basis of the performance differences observed between couples of decision alternatives on the given family of criteria functions.

Let us construct, for instance, a random bipolar-valued outranking digraph with seven decision actions denotes $a1$, $a2$, ..., $a7$. We need therefore to first generate a corresponding random performance tableaux (see Listing 1.1 below).

Listing 1.1 Generating a random performance tableau

```
>>> from outrankingDigraphs import *
>>> pt = RandomPerformanceTableau(numberOfActions=7,\
...                                 seed=100)
>>> pt
   *------- PerformanceTableau instance description ------*
    Instance class     : RandomPerformanceTableau
    Seed               : 100
    Instance name      : randomperftab
    Actions            : 7
    Criteria           : 7
    NaN proportion (%) : 6.1
>>> pt.showActions()
  *----- show digraphs actions --------------*
   key:  a1
    name:       action #1
    comment:    RandomPerformanceTableau() generated.
   key:  a2
    name:       action #2
    comment:    RandomPerformanceTableau() generated.
     ...
     ...
   key:  a7
    name:       action #7
    comment:    RandomPerformanceTableau() generated.
```

In this example we consider furthermore a family of seven **equisignificant cardinal criteria functions** $g1$, $g2$, ..., $g7$, measuring the performance of each alternative on a rational scale from 0.0 (worst) to 100.00 (best). In order to capture the grading procedure's potential uncertainty and imprecision, each criterion function $g1$ to $g7$ admits three performance **discrimination thresholds** of 2.5, 5.0 and 80.0 pts for warranting respectively any indifference, preference or considerable performance difference situation.

Listing 1.2 Inspecting the performance criteria

```
>>> pt.showCriteria()
  *----  criteria -----*
   g1 'RandomPerformanceTableau() instance'
     Scale = [0.0, 100.0]
     Weight = 1.0
     Threshold ind : 2.50 + 0.00x ; percentile: 4.76
     Threshold pref : 5.00 + 0.00x ; percentile: 9.52
     Threshold veto : 80.00 + 0.00x ; percentile: 95.24
   g2 'RandomPerformanceTableau() instance'
     Scale = [0.0, 100.0]
     Weight = 1.0
     Threshold ind : 2.50 + 0.00x ; percentile: 6.67
     Threshold pref : 5.00 + 0.00x ; percentile: 6.67
     Threshold veto : 80.00 + 0.00x ; percentile: 100.00
```

```
15       ...
16       ...
17    g7 'RandomPerformanceTableau() instance'
18      Scale = [0.0, 100.0]
19      Weight = 1.0
20      Threshold ind : 2.50 + 0.00x ; percentile: 0.00
21      Threshold pref : 5.00 + 0.00x ; percentile: 4.76
22      Threshold veto : 80.00 + 0.00x ; percentile: 100.00
```

On criteria function $g1$ (see Lines Listing 1.2 6-8 above) we observe, for instance, about 5% of *indifference*, about 90% of *preference* and about 5% of *considerable* performance difference situations. The individual performance evaluation of all decision alternative on each criterion are gathered in a **performance tableau**.

Listing 1.3 Inspecting the performance table

```
1  >>> pt.showPerformanceTableau()
2      *----  performance tableau -----*
3      criteria |  'a1'  'a2'  'a3'  'a4'  'a5'  'a6'  'a7'
4      ---------|------------------------------------------
5       'g1'    |  15.2  44.5  57.9  58.0  24.2  29.1  96.6
6       'g2'    |  82.3  43.9   NA   35.8  29.1  34.8  62.2
7       'g3'    |  44.2  19.1  27.7  41.5  22.4  21.5  56.9
8       'g4'    |  46.4  16.2  21.5  51.2  77.0  39.4  32.1
9       'g5'    |  47.7  14.8  79.7  67.5   NA   90.7  80.2
10      'g6'    |  69.6  45.5  22.0  33.8  31.8   NA   48.8
11      'g7'    |  82.9  41.7  12.8  21.9  75.7  15.4   6.0
```

It is noteworthy to mention the three **missing data** (`NA`) cases: action $a3$ is missing, for instance, a grade on criterion $g2$ (see Listing 1.3 Line 6 above).

1.2 The bipolar-valued outranking digraph

Given the previous random performance tableau *pt*, the `BipolarOutrankingDigraph` constructor computes the corresponding **bipolar-valued outranking digraph**.

Listing 1.4 Example of random bipolar-valued outranking digraph

```
>>> g = BipolarOutrankingDigraph(pt)
>>> g
  *------- Object instance description ------*
   Instance class       : BipolarOutrankingDigraph
   Instance name        : rel_randomperftab
   Actions              : 7
   Criteria             : 7
   Size                 : 20
   Determinateness (%)  : 63.27
   Valuation domain     : [-1.00;1.00]
   Attributes           : [
        'name', 'actions',
        'criteria', 'evaluation', 'NA',
        'valuationdomain', 'relation',
```

```
15         'order', 'gamma', 'notGamma', ...
16         ]
```

The resulting digraph contains 20 positive (valid) outranking realtions. And, the mean majority criteria significance support of all the pairwise outranking situations is 63.3% (see Listing 1.4 Lines 8-9). We may inspect the complete $[-1.0,+1.0]$-valued adjacency table as follows.

Listing 1.5 Inspecting the valued adjacency table

```
1  >>> odg.showRelationTable()
2    * ---- Relation Table -----
3     r(x,y)|  'a1'   'a2'   'a3'   'a4'   'a5'   'a6'   'a7'
4     ------|--------------------------------------------------
5      'a1' | +1.00  +0.71  +0.29  +0.29  +0.29  +0.29  +0.00
6      'a2' | -0.71  +1.00  -0.29  -0.14  +0.14  +0.29  -0.57
7      'a3' | -0.29  +0.29  +1.00  -0.29  -0.14  +0.00  -0.29
8      'a4' | +0.00  +0.14  +0.57  +1.00  +0.29  +0.57  -0.43
9      'a5' | -0.29  +0.00  +0.14  +0.00  +1.00  +0.29  -0.29
10     'a6' | -0.29  +0.00  +0.14  -0.29  +0.14  +1.00  +0.00
11     'a7' | +0.00  +0.71  +0.57  +0.43  +0.29  +0.00  +1.00
12    Valuation domain: [-1.0; 1.0]
```

Considering the given performance tableau *pt*, the `BipolarOutrankingDigraph` class constructor computes the characteristic value $r(x,y)$ of a *pairwise outranking* relation $x \succsim y$ (Bisdorff [2013], Bisdorff [2020]) in a default *normalised* **valuation domain** $[-1.0,+1.0]$ with the *median* value 0.0 acting as **indeterminate** characteristic value. The semantics of $r(x,y)$ are the following.

1. When $r(x,y) > 0.0$, it is more *True* than *False* that x **outranks** y, i.e. alternative x is at least as well performing than alternative y on a weighted majority of criteria **and** there is no considerable negative performance difference observed in disfavour of x,
2. When $r(x,y) < 0.0$, it is more *False* than *True* that x **outranks** y, i.e. alternative x is **not** at least as well performing on a weighted majority of criteria than alternative y **and** there is no considerable positive performance difference observed in favour of x,
3. When $r(x,y) = 0.0$, it is **indeterminate** whether x outranks y or not.

1.3 Pairwise comparisons

From above given semantics, we may consider (see Listing 1.5 Line 5 above) that $a1$ outranks $a2$ ($r(a1,a2) > 0.0$), but not a_7 ($r(a1,a7) = 0.0$). In order to comprehend the characteristic values shown in the relation table above, we may furthermore inspect the details of the pairwise multiple criteria comparison between alternatives $a1$ and $a2$.

Listing 1.6 Inspecting a pairwise multiple criteria comparison

```
>>> odg.showPairwiseComparison('a1','a2')
```

```
 *------------  pairwise comparison ----*
  Comparing actions : (a1, a2)
  crit. wght.  g(x)  g(y)    diff  | ind   pref    r()
  -------------------------------        --------------------
   g1   1.00  15.17  44.51  -29.34 | 2.50  5.00   -1.00
   g2   1.00  82.29  43.90  +38.39 | 2.50  5.00   +1.00
   g3   1.00  44.23  19.10  +25.13 | 2.50  5.00   +1.00
   g4   1.00  46.37  16.22  +30.15 | 2.50  5.00   +1.00
   g5   1.00  47.67  14.81  +32.86 | 2.50  5.00   +1.00
   g6   1.00  69.62  45.49  +24.13 | 2.50  5.00   +1.00
   g7   1.00  82.88  41.66  +41.22 | 2.50  5.00   +1.00
  ----------------------------------------
  Valuation in range: -7.00 to +7.00; r(x,y): +5/7 = +0.71
```

The outranking characteristic value $r(a1 \succsim a2)$ represents the **majority margin** resulting from the difference between the weights of the criteria in favor and the weights of the criteria in disfavor of the statement that alternative $a1$ is at least as well performing as alternative $a2$. No considerable performance difference being observed above, no veto or counter-veto situation is triggered in this pairwise comparison. Such a situation is, however, observed for instance when we pairwise compare the performances of alternatives $a1$ and $a7$.

```
>>> odg.showPairwiseComparison('a1','a7')
 *------------  pairwise comparison ----*
  Comparing actions : (a1, a7)
  crit. wght.  g(x)  g(y)    diff  | ind   pref    r()   |  v
          veto
  -------------------------------   ------------------
      -----------
   g1   1.00  15.17  96.58  -81.41 | 2.50  5.00   -1.00 | 80.00
     -1.00
   g2   1.00  82.29  62.22  +20.07 | 2.50  5.00   +1.00 |
   g3   1.00  44.23  56.90  -12.67 | 2.50  5.00   -1.00 |
   g4   1.00  46.37  32.06  +14.31 | 2.50  5.00   +1.00 |
   g5   1.00  47.67  80.16  -32.49 | 2.50  5.00   -1.00 |
   g6   1.00  69.62  48.80  +20.82 | 2.50  5.00   +1.00 |
   g7   1.00  82.88   6.05  +76.83 | 2.50  5.00   +1.00 |
  ----------------------------------------
  Valuation in range: -7.00 to +7.00; r(x,y)= +1/7 => 0.0
```

This time, we observe a 57.1% majority of criteria significance $[(1/7+1)/2 = 0.571]$ warranting an *as well as performing* situation. Yet, we also observe a considerable negative performance difference on criterion $g1$ (see Listing 1.5 first row in the relation table above). Both contradictory facts trigger eventually an *indeterminate* outranking situation [BIS-2013].

1.4 Recoding the digraph valuation

All outranking digraphs, being of root type `Digraph`, inherit the methods available under this latter class. The characteristic valuation domain of a digraph may, for instance, be recoded with the `recodeValutaion()` method below to the *integer* range $[-7, +7]$, i.e. plus or minus the global significance of the family of criteria considered in this example instance.

Listing 1.7 Recoding the digraph valuation

```
>>> odg.recodeValuation(-37,+37)
>>> odg.valuationdomain['hasIntegerValuation'] = True
>>> Digraph.showRelationTable(odg,ReflexiveTerms=False)
  * ---- Relation Table -----
   r(x,y)  |  'a1'  'a2'  'a3'  'a4'  'a5'  'a6'  'a7'
  ---------|------------------------------------------
    'a1'   |    0     5     2     2     2     2     0
    'a2'   |   -5     0    -1    -1     1     2    -4
    'a3'   |   -1     2     0    -1    -1     0    -1
    'a4'   |    0     1     4     0     2     4    -3
    'a5'   |   -1     0     1     0     0     2    -1
    'a6'   |   -1     0     1    -1     1     0     0
    'a7'   |    0     5     4     3     2     0     0
    Valuation domain: [-7;+7]
```

Notice that the reflexive self comparison characteristic $r(x,x)$ is set above by default to the median indeterminate valuation value 0; the reflexive terms of binary relation being generally ignored in most of the DIGRAPH3 3 resources.

1.5 The strict outranking digraph

From the theory (Bisdorff [2013], Bisdorff [2020]) we know that a bipolar-valued outranking digraph is **weakly complete**, i.e. if $r(x,y) < 0.0$ then $r(y,x) \geq 0.0$. From this property follows that a bipolar-valued outranking relation verifies the **coduality** principle: the **dual** (*strict negation* $-$ [1]) of the **converse** (*inverse* $\approx$) of the outranking relation corresponds to its *strict outranking* part.

We may visualize the **codual** (*strict*) outranking digraph with a graphviz drawing [2].

```
>>> cdodg = -(~odg)
>>> cdodg.exportGraphViz('codualOdg')
  *---- exporting a dot file for GraphViz tools ---------*
   Exporting to codualOdg.dot
   dot -Grankdir=BT -Tpng codualOdg.dot -o codualOdg.png
```

Many more tools for exploiting bipolar-valued outranking digraphs are available in the DIGRAPH3 3 resources (see the technical documentation of the `outrankingDigraphs` module and the `perfTabs` module).

[1] Not to be confused with the dual graph of a plane graph g that has a vertex for each face of g. Here we mean the *less than* (strict converse) relation corresponding to a *greater or equal* relation, or the *less than or equal* relation corresponding to a (strict) *better than* relation.

[2] The `exportGraphViz()` method is depending on drawing tools from the graphviz software (https://graphviz.org/). On Linux Ubuntu or Debian you may try `sudo apt-get install graphviz` to install them. There are ready *dmg* installers for Mac OSX.

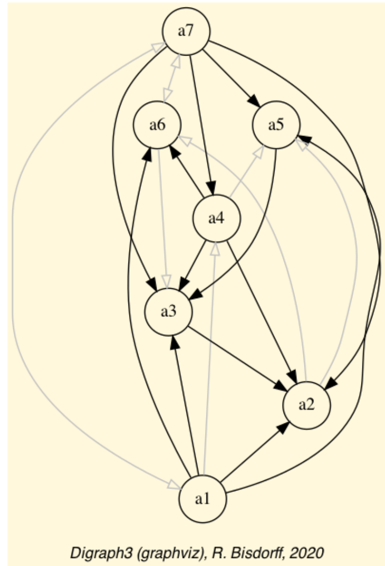

Fig. 1.1 The codual outranking digraph. It becomes readily clear now from the picture above that both alternatives $a1$ and $a7$ are *not outranked* by any other alternatives. Hence, $a1$ and $a7$ appear as *weak* CONDORCET winner and may be recommended as potential *best decision actions* in this illustrative preference modelling exercise.

References

[Bisdorff 2013] BISDORFF, R.: On Polarizing Outranking Relations with Large Performance Differences. In: *Journal of Multi-Criteria Decision Analysis, Wiley* 20 (2013), S. 3–12. – URL http://hdl.handle.net/10993/245 4, 6

[Bisdorff 2020] BISDORFF, R.: *Best multiple criteria choice: the Rubis outranking method. MICS Algorithmic Decision Theory Course Lecture 7, University of Luxembourg*. 2020 4, 6

Appendix A
Appendix

Use the template *appendix.tex* together with the Springer document class SVMono (monograph-type books) or SVMult (edited books) to style appendix of your book in the Springer layout.