PARSEC: An R Package for Partial Orders in Socio-Economics

Alberto Arcagni

1 Introduction

The R (R Core Team 2015) package parsec provides tools for multidimensional evaluation with ordinal variables. In particular its functions allow the application of the poset-based approach (Fattore et al. 2011a,b, 2012) and of the counting approach (Alkire and Foster 2011a,b). The package is available on CRAN (the Comprehensive R Archive Network).

Recently, Fattore published an article where he describes in detail the poset-based approach and specifies how to include external information on attribute relevance (Fattore 2015). Therefore parsec has been upgraded by introducing functions to generate more elaborated posets so that they represent such external informations.

The basic functions of the package are described in a chapter of the book "Multi-indicator Systems and Modelling in Partial Order" (Fattore and Arcagni 2014) therefore this work focuses on the new functionalities. In this work it is assumed that the reader has a basic knowledge of the methodology, of the R language and of the object oriented programming. The work is divided into two parts.

Section 2 describes how to create incidence matrices that are the main structures used in parsec to model posets. Observe that some methods to generate incidence matrices require to distinguish between *achievement posets* and *attribute posets* (Fattore 2015). The first ones are the posets used to define the comparabilities between profiles, whereas attribute posets are necessary to include the external information on attribute relevance. The package provides various functions to create incidence matrices, and they are exposed in the following sections. Section 2.1 illustrates how to check if a generic boolean matrix can be utilized as incidence

A. Arcagni (✉)
Department of Statistics and Quantitative Methods, University of Milano-Bicocca, Milan, Italy
e-mail: alberto.arcagni@unimib.it

M. Fattore, R. Bruggemann (eds.), *Partial Order Concepts in Applied Sciences*,
DOI 10.1007/978-3-319-45421-4_19

matrix. Section 2.2 describes the functions useful to generate a *basic* achievement poset. Section 2.3 focuses on functions for the creation of attribute posets and for the generation of achievement posets derived by them.

Section 3 represents the second part of this work. Once the achievement poset is realized it can be used to evaluate the indicators provided by the poset-based approach. Such section describes the basic use of the `evaluation` function. Functions are exemplified by four achievement poset, then Sect. 3.1 concludes the examples by proposing a comparison of the results.

Section 4 is devoted to conclusions.

2 Incidence Matrices and Posets

A partial order set, *poset*, is a set X equipped with a partial order relation $\leq$ (for details see Fattore 2015). In `parsec` posets are represented through *incidence matrices*. They are objects of class `incidence` that are boolean matrices whose rows and columns are named in order to list the elements of the set. Boolean values assumed by the incidence matrix at line `i` and column `j` indicate if the `i`th element of the set is lower or equal to the `j`th one.

Through examples, it is illustrated how to construct an incidence matrix. In the package there are functions:

- to check if a boolean matrix can be used as incidence matrix;
- to generate incidence matrices of basic achievement posets;
- to generate incidence matrices of attribute posets;
- to generate incidence matrices of achievement posets starting from an attribute poset.

2.1 Check Boolean Matrices

First of all, we show how to check if a boolean matrix can be used as incidence matrix. By definition, a partial order relation satisfies the properties of reflexivity, antisymmetry and transitivity. Let the set $X = \{x, y, z\}$ represent it through a character vector and create the named boolean matrix `I` that will store the partial order relation $\leq$.

```
X <- c("x", "y", "z")
I <- matrix(FALSE, 3, 3)
rownames(I) <- colnames(I) <- X
I
#       x     y     z
# x FALSE FALSE FALSE
# y FALSE FALSE FALSE
# z FALSE FALSE FALSE
```

Reflexivity property states that $x \le x$ for all $x \in X$. It can be obtained by setting the diagonal of matrix I all equal to TRUE. Function reflexivity checks if the matrix satisfies such property.

```
diag(I) <- TRUE; I
#       x     y     z
# x  TRUE FALSE FALSE
# y FALSE  TRUE FALSE
# z FALSE FALSE  TRUE
reflexivity(I)
# [1] TRUE
```

Antisymmetry property states that if $x \le y$ and $y \le x$ then $x = y$, for all $x, y \in X$. In the following example we set x lower or equal to y and y lower or equal to x. Since they are represented in two different rows and columns of matrix I they cannot be the same element of the set, therefore the function antisymmetry returns FALSE. Then we have to set y greater than x in order to get an antisymmetric relation.

```
I["x", "y"] <- TRUE; I["y", "x"] <- TRUE
antisymmetry(I)
# [1] FALSE
I["y", "x"] <- FALSE; I
#       x     y     z
# x  TRUE  TRUE FALSE
# y FALSE  TRUE FALSE
# z FALSE FALSE  TRUE
antisymmetry(I)
# [1] TRUE
```

Transitivity states that if $x \le y$ and $y \le z$, then $x \le z$, $x, y, z \in X$. Therefore, if I["y", "z"] is set TRUE the function transitivity(I) returns FALSE because, in the previous step, I["x", "y"] is TRUE. Starting from matrix I, there are two choices to obtain boolean matrices I1 and I2 that can be used as incidence matrices where z and y can be compared:

- matrix I1: set I1["z", "y"] equal to TRUE, instead of I1["y", "z"];
- matrix I2: set TRUE both I2["y", "z"] and I2["x", "z"].

```
I1 <- I
I1["z", "y"] <- TRUE
transitivity(I1)
# [1] TRUE
I2 <- I
I2["y", "z"] <- TRUE
I2["x", "z"] <- TRUE
transitivity(I2)
# [1] TRUE
```

Once the boolean matrix satisfies all the required properties (function `is.partialorder` checks all of them) its class can be set to `incidence`. Then the matrix represents a poset and it can be used with methods and functions of the package. For instance, the method of function `plot` associated by the package to the class `incidence` returns a Hasse-diagram instead of a scatter-plot. Hasse-diagram is an oriented graph where the edge directions are from top to bottom and represents the cover relation.

```
is.partialorder(I)
# [1] TRUE
class(I) <- "incidence"; I
#       x     y     z
# x  TRUE  TRUE FALSE
# y FALSE  TRUE FALSE
# z FALSE FALSE  TRUE
# attr(,"class")
# [1] "incidence"
plot(I, main = "I")

is.partialorder(I1)
# [1] TRUE
class(I1) <- "incidence"; I1
#       x    y     z
# x  TRUE TRUE FALSE
# y FALSE TRUE FALSE
# z FALSE TRUE  TRUE
# attr(,"class")
# [1] "incidence"
plot(I1, main = "I1")

is.partialorder(I2)
# [1] TRUE
class(I2) <- "incidence"; I2
#       x     y    z
# x  TRUE  TRUE TRUE
# y FALSE  TRUE TRUE
# z FALSE FALSE TRUE
# attr(,"class")
# [1] "incidence"
plot(I2, main = "I2")
```

The images generated by the `plot` commands are shown in Fig. 1. From Fig. 1 the partial order relations described in this section can be summarized. Apart reflexivity, matrix `I` only states that $x \leq y$, i.e., only `I["x", "y"]` is `TRUE`. This can be observed in the corresponding Hasse-diagram where there is only the edge from y to x. Observe that z cannot be compared with any other element of the set X.

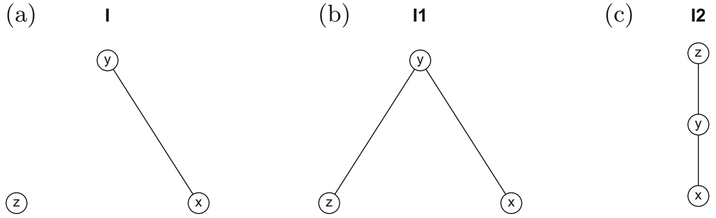

Fig. 1 Hasse-diagrams of partial orders described by incidence matrices `I`, `I1` and `I2`

On the contrary, in the Hasse-diagram of `I1` matrix it can be observed that z is lower than y because `I1["z", "y"]` has been set `TRUE`. Matrix `I2` is similar to `I` but `I2["y", "z"]` is `TRUE`. From the corresponding Hasse-diagram it can be observed the reason why by transitivity $x \leq z$ and `I2["x", "z"]` has to be `TRUE`. Matrix `I2` represents a complete order.

2.2 *The Basic Achievement Poset*

In multidimensional measurement, a set of k ordinal variables (they are also called *attributes*) $v_1, \dots, v_k$ may be identified to describe a complex phenomena. Let the variable v_j have m_j degrees, $j = 1, \dots, k$. All the combinations of their modalities represent the set of *achievement profiles*, denoted by Π. Without introducing any assumption, the *basic achievement poset* can be obtained simply comparing the scores that each variable assumes in a profile. The following example shows how to obtain in `parsec` the basic achievement poset from a set of $k = 3$ variables with $m_1 = 3, m_2 = 2$ and $m_3 = 4$.

Variables can be simply defined through the vector of their degrees. It can be named a vector in order to assign at each variable a label. Then the function `var2prof` generates the set of achievement profiles Π.

```
m <- c(x=3, y=2, z=4)
PI <- var2prof(varlen = m); PI
# $profiles
#     x y z
# 111 1 1 1
# 211 2 1 1
# ...
# 224 2 2 4
# 324 3 2 4
#
# $freq
# 111 211 311 121 221 321 112 212 312 122 222 322 113 213
# 1   1   1   1   1   1   1   1   1   1   1   1   1   1
# 313 123 223 323 114 214 314 124 224 324
# 1   1   1   1   1   1   1   1   1   1
#
# attr(,"class")
# [1] "wprof"
```

Function `var2prof` returns an object of class `wprof` composed by a `data.frame` representing the set of achievement profiles Π and a vector of frequencies associated with each profile, that can be modified and it is useful to evaluate synthetic measures whenever the set of achievement profiles is associated with a population.

An object of class `wprof` can be used as argument of function `getzeta` which returns the incidence matrix of the basic achievement poset (generally identified with letter Z). Such function returns an object of class `incidence` that can be used as shown before.

```
Z <- getzeta(PI)
plot(Z)
```

The output of `plot` command is shown in Fig. 2.

2.3 External Information on Attribute Relevance

How to obtain the basic achievement poset is described in Sect. 2.2 and we pointed out that it is the achievement poset that is defined simply by comparing the scores that each attribute assumes in profiles. But it is also possible to introduce external informations on attribute relevance. It can be done through an *attribute poset*,

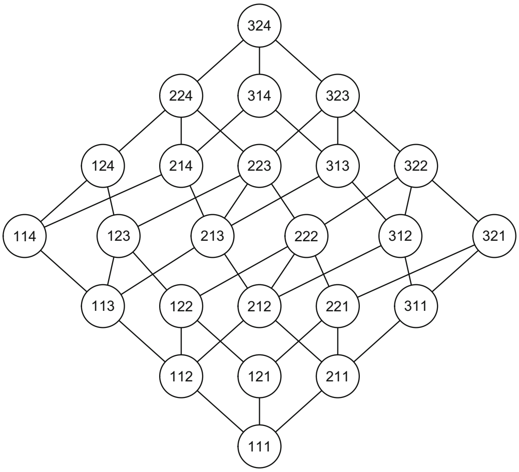

Fig. 2 Basic achievement poset

generally identified as Λ. The function `getlambda` has been recently introduced in the package to define the attribute poset and it generates an incidence matrix simply by the definition of comparabilities through attributes. For instance, the incidence matrices defined in Sect. 2.1 can be obtained with function `getlambda` simply by declaring comparabilities and by listing the attributes that are not comparable with the others.

```
I  <- getlambda(x < y, z)
I1 <- getlambda(x < y, z < y)
I2 <- getlambda(x < y, z > y)
```

Once the attribute poset is defined it is necessary to get its set of linear extensions $\Omega(\Lambda)$ (see Fattore 2015). Function `LE` returns the linear extensions of a poset. The output is a list of character vectors. Each vector represents the attribute names from the lowest to the highest one.

```
Omega_I <- LE(I);
# $LE1
# [1] "x" "y" "z"
#
# $LE2
# [1] "x" "z" "y"
#
# $LE3
# [1] "z" "x" "y"

Omega_I1 <- LE(I1); Omega_I1
# $LE1
# [1] "x" "z" "y"
#
# $LE2
# [1] "z" "x" "y"

Omega_I2 <- LE(I2); Omega_I2
# $LE1
# [1] "x" "y" "z"
```

Observe that matrix `I2` represents a complete order therefore it has only a linear extension. Function `mrg` generates the incidence matrix of the achievement poset from $\Omega(\Lambda)$ and from the attribute definitions. Consider, for instance, the attributes defined in Sect. 2.2. The achievement posets corresponding to each attribute poset `I`, `I1` and `I2` are shown in Fig. 3 and the code to get them follows:

```
m <- c(x=3, y=2, z=4)

Z_I <- mrg(Omega_I, varlen = m)
plot(Z_I, shape = "equispaced")
```

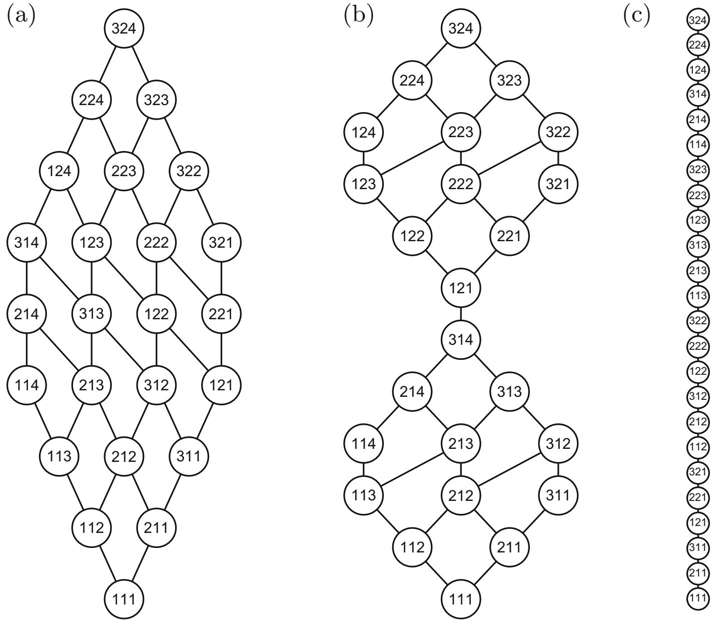

Fig. 3 Achievement poset obtained from attribute posets (**a**) `I`, (**b**) `I1` and (**c**) `I2`

```
Z_I1 <- mrg(Omega_I1, varlen = m)
plot(Z_I1, shape = "equispaced")

Z_I2 <- mrg(Omega_I2, varlen = m)
plot(Z_I2)
```

In conclusion of this section, observe that the basic achievement poset shown in Fig. 2 can also be obtained through the `getlambda` function by listing the attributes names without introducing any relation between them. In this case, the set $\Omega(\Lambda)$ is represented by all the permutations of attributes names.

```
I0 <- getlambda(x, y, z)
Omega_I0 <- LE(I0);Omega_I0
# $LE1
# [1] "x" "y" "z"
#
# $LE2
# [1] "x" "z" "y"
#
# $LE3
# [1] "y" "x" "z"
```

```
#
# $LE4
# [1] "y" "z" "x"
#
# $LE5
# [1] "z" "x" "y"
#
# $LE6
# [1] "z" "y" "x"
m <- c(x=3, y=2, z=4)
Z <- mrg(Omega_I0, varlen = m)
plot(Z)
```

3 From the Achievement Poset to the Incidence Function

In Sect. 2, four achievement posets were proposed for attributes x, y and z: `Z`, `Z_I`, `Z_I1` and `Z_I2`. Once an achievement poset is available, the choice of a *threshold* allows the application of the poset-based methodology in order to obtain an *incidence function*. Details about thresholds selections are available in Fattore et al. (2011a, 2012) and Fattore (2015), here we recall only that the threshold is an antichain of the poset whose downset identifies the profiles that certainly represent a condition of deprivation.

In `parsec` a threshold is a vector whose elements allow to identify rows/-columns of the incidence matrix. Rows and columns of the incidence matrix can be identified through a boolean vector or a vector of profiles. Function `evaluation` applies the poset-based methodology starting from a threshold and an object of class `incidence` representing the achievement poset. It is also possible to add an object of class `wprof` in order to define the profile frequency distribution to get the synthetic measures in relation to a population.

In the following the results of function `evaluation` are shown through examples starting from the incidence matrices `Z`, `Z_I`, `Z_I1` and `Z_I2` defined before.

Profiles `122`, `221` and `113` are used as threshold proposal. The function `downset` returns the downset of a subset of profiles. If the subset of profiles is not an antichain it means that one or more profiles are not necessary to generate such downset. The function `gen.downset` returns the antichain that generates the input downset. Observe that antichains depend on the incidence relation, therefore the threshold proposal cannot be an antichain even if such profiles are not comparable in the basic achievement poset. Therefore, for each example the threshold proposal is analysed.

First of all, consider the basic achievement poset.

```
proposal <- c("122", "221", "113")

dwn <- downset(Z, proposal); which(dwn)
# 111 211 121 221 112 122 113
#   1   2   4   5   7  10  13

threshold <- gen.downset(Z, dwn); which(threshold)
# 221 122 113
#   5  10  13
```

The downset of the threshold proposal in this case is represented by profiles `111`, `211`, `121`, `221`, `112`, `122` and `113`. The antichain that generates this downset is equal to the proposal, therefore it is used as threshold.

```
results <- evaluation(threshold = threshold, zeta = Z)
```

The `evaluation` function returns an object of class `parsec`. Such class of objects is a list summarizing the input informations and the results. In particular, in the results are provided the distributions of different indices computed by uniform sampling of the linear extensions of the poset through a `C` implementation of the Bubley–Dyer algorithm (Bubley and Dyer 1999).

Methods for the class `parsec` has been associated with functions `summary` and `plot`. The `evaluation` function returns a large variety of results useful to get synthetic measures, plots and to develop further analysis. Therefore it may be useful to summarize its results. Function `summary` returns a data-frame that for each profile provides

weights its weight in population;
threshold the boolean variable indicating if it belongs to the threshold;
id. function the corresponding value of the identification function, i.e., the fraction of sampled linear extensions where the profile is in the downset of the threshold;
average rank the average of the ranks that the profile assumes in each of the sampled linear extension;
abs. severity (absolute severity) the average graph distance of the profile from the first one above all threshold elements (the distance is set equal to 0 for profiles above the threshold);
rel. severity (relative severity) equal to the absolute severity divided by its maximum, that is, the absolute severity of the minimal element in the linear extension;
abs. wealth gap (absolute wealth gap) the average graph distance from the maximum threshold element (the distance is set equal to 0 for profiles in the downset of threshold elements);
rel. wealth gap (relative wealth gap) equal to the absolute wealth gap divided by its maximum, that is, the absolute distance of the threshold from the maximal element in the linear extension.

The `evaluation` function returns the synthetic indicators `poverty_gap` and `wealth_gap` that are, respectively, the weighted means of the relative severity and of the relative wealth gap. The code to get the summary of the results and the plots follows:

```
sry_results <- summary(results); str(sry_results)
# 'data.frame': 24 obs. of  8 variables:
# $ weights        : num  1 1 1 1 1 1 1 1 1 1 ...
# $ threshold      : logi  FALSE FALSE FALSE TRUE ...
# $ id. function   : num  1 1 1 1 1 ...
# $ average rank   : num  1 4.54 6.2 7.36 8.84 ...
# $ abs. severity  : num  9.95 7.73 7.06 2.45 8 ...
# $ rel. severity  : num  1 0.769 0.708 0.251 0.797 ...
# $ abs. wealth gap: num  0 0 0 0 0 ...
# $ rel. wealth gap: num  0 0 0 0 0 ...

results$poverty_gap
# [1] 0.2736463
results$wealth_gap
# [1] 0.4427691

plot(results)
```

Figure 4 shows the images provided by function `plot`. Figure 4a highlights threshold profiles in the Hasse-diagram of the achievement poset. Figure 4b shows the relative frequencies of the times a profile is threshold in the sampled linear extensions of the achievement poset, therefore it is a representation of the relevance of the threshold profiles. The identification function is shown in Fig. 4c. Finally, Fig. 4d shows a representation of the rank distributions of each profile in the sampled linear extensions. The grey scale used to fill the rectangles represents the rank value (white means rank equal to 1, black means maximum rank). The height of the rectangles represents the corresponding relative frequency. Therefore, to each profile corresponds a bar of height 1, and a dark bar means that the profile tends to assume higher ranks.

3.1 Final Comparisons

In conclusion a comparison of the results obtained with the different achievement posets is proposed.

It may be of interest a comparison on the threshold variations induced by the introduction of external informations on attribute relevance. Previously we defined a threshold proposal that in the basic achievement poset is an antichain therefore it is suitable to be used as threshold. We also observed that the introduction of external informations on attribute relevance changes the achievement poset, consequentially the threshold proposal may not be an antichain. Therefore we proposed to evaluate

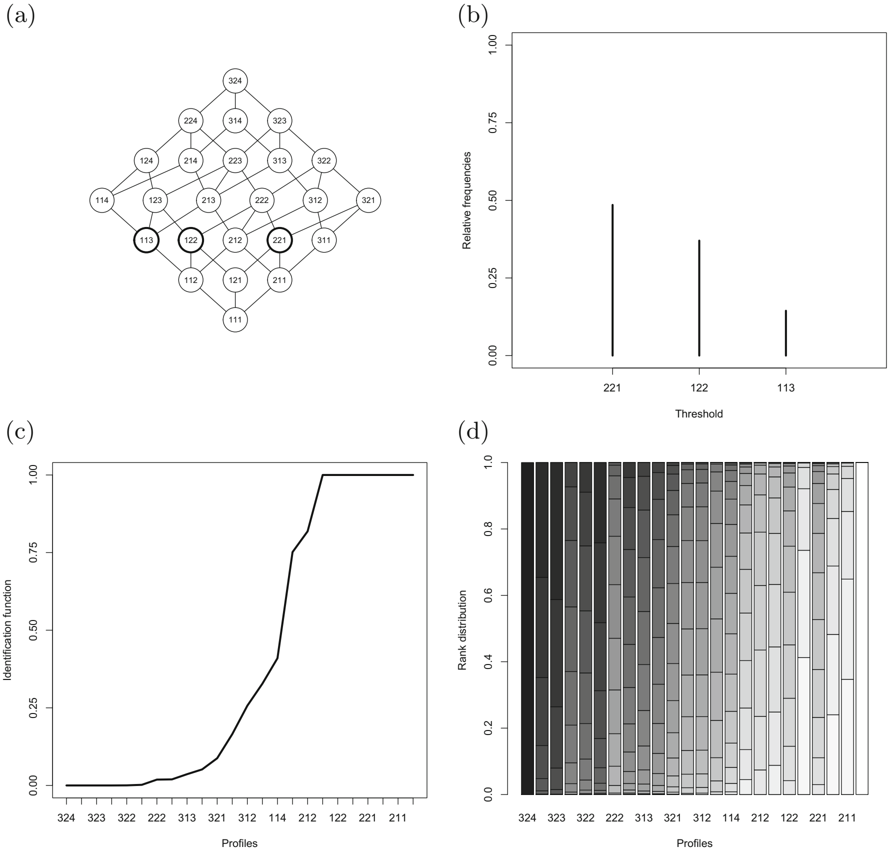

Fig. 4 Plots of the results provided by the `evaluation` function. (**a**) Threshold in the Hasse-diagram. (**b**) Relevance of threshold profiles. (**c**) Identification function. (**d**) Profiles rank distributions

the proposal downset and then to identify the antichain that generates such downset. Here below it is proposed the code to apply this procedure to the achievement posets generated by the attribute posets `I`, `I1` and `I2`.

```
dwn <- downset(Z_I, proposal); which(dwn)
# 111 211 311 121 221 112 212 312 122 113
#   1   2   3   4   5   7   8   9  10  13
threshold_I <- gen.downset(Z_I, dwn);
which(threshold_I)
# 221 122 113
#   5  10  13

dwn <- downset(Z_I1, proposal); which(dwn)
# 111 211 311 112 212 312 113 213 313 114 214 314
#   1   2   3   4   5   6   7   8   9  10  11  12
# 121 221 122
#  13  14  16
threshold_I1 <- gen.downset(Z_I1, dwn);
which(threshold_I1)
# 221 122
#  14  16

dwn <- downset(Z_I2, proposal); which(dwn)
# 111 211 311 121 221 321 112 212 312 122 222 322
#   1   2   3   4   5   6   7   8   9  10  11  12
# 113
#  13
threshold_I2 <- gen.downset(Z_I2, dwn);
which(threshold_I2)
# 113
#  13
```

In the case of the achievement poset generated by the attribute poset `I`, Figs. 3a and 1a, the threshold proposal still remains an antichain.

When the attribute *y* is more important than attributes *x* and *z*, attribute poset `I1` shown in Fig. 1b, the profile `113` is lower than profiles `122` and `221`, Fig. 3b, because, even if its score on attribute *z* is 3, it is less important than the score it assumes on attribute $y = 1$, which is lower than the scores of *y* in the other two profiles. Therefore it is not necessary to introduce profile `113` into the threshold, as it can be observed in the commented results of the code proposed before.

On the contrary, in the achievement poset (Fig. 3c) obtained by the attribute poset `I2` (Fig. 1c) profile `113` is the only one of the proposed one that is useful as threshold because attribute *z* is the most relevant.

For the sake of brevity, only the comparison between the synthetic measures, Table 1, and the inequality curves, Fig. 5, are shown.

For all the examples the poverty gap is lower than the wealth gap, this is consequence of the fact that profiles in the threshold proposal are closer to the bottoms of the different achievement posets than to their tops. Moreover the tendency of

Table 1 Comparison of synthetic measures between different achievement posets obtained through different attribute posets and with same threshold proposal

Attribute poset	Poverty gap	Wealth gap
Basic	0.2736	0.4428
`I`	0.3425	0.4393
`I1`	0.4528	0.5388
`I2`	0.5385	0.5455

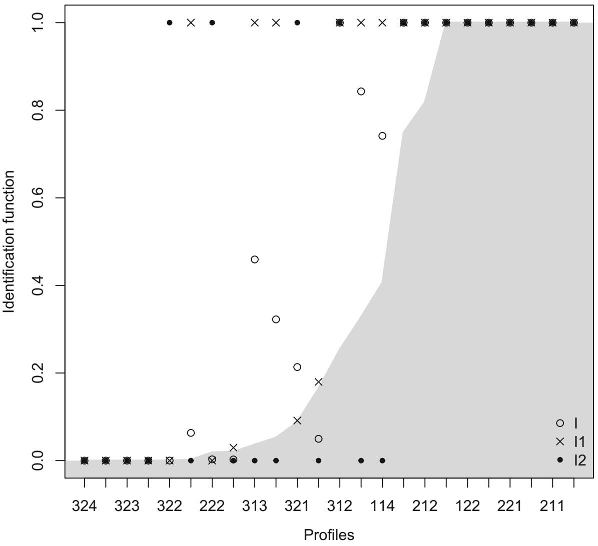

Fig. 5 Comparison of identification functions between different achievement posets obtained through different attribute posets and with same threshold proposal

such gaps to increase from the basic achievement poset to the achievement poset based on `I2` may be a consequence of the evolution of the examples, obtained by the progressive introduction of external information. This procedure increased the achievement posets levels and reduced the threshold size increasing the relative distances of the profiles from the threshold.

Generally, identification functions are represented in increasing order. In Fig. 5 we preferred to maintain the profile order obtained with the identification function generated by the basic poset (the gray background) and to show the other identification functions through different point characters. In the last example, based on `I2`, the achievement poset is a complete order, therefore the identification function assumes only two values: 1 when the profile is lower or equal than the threshold `113`, 0 otherwise.

4 Conclusion

Some examples were presented in order to expose the new functions of `parsec`, from poset definitions by incidence matrices to the analysis of the results provided by the main functions of the package. In particular, this work focuses on the introduction of external informations on attribute relevance, it shows that function `getlambda` simplifies the creation of posets and that function `LE` generates their linear extensions. Then, from the linear extensions and from the attribute definitions, it has been shown how to get achievement poset with function `mrg`. In the second part, we suggested how to choose a threshold, starting from a proposal and by its analysis in a specific achievement poset. Then we compared the main results of the function `evaluation`.

In conclusion we recall that the package `parsec` is available on CRAN and that it is still in development in order to add new functionalities and to optimize its procedures.

References

Alkire S, Foster J (2011a) Counting and multidimensional poverty measurement. J Public Econ 95(7):476–487

Alkire S, Foster J (2011b) Understandings and misunderstandings of multidimensional poverty measurement. J Econ Inequal 9(2):289–314

Bubley R, Dyer M (1999) Faster random generation of linear extensions. Discret Math 201(1): 81–88

Fattore M (2015) Partially ordered sets and the measurement of multidimensional ordinal deprivation. Soc Indic Res 1–24

Fattore M, Arcagni A (2014) Parsec: an R package for poset-based evaluation of multidimensional poverty. In: Multi-indicator systems and modelling in partial order. Springer, Berlin, pp 317–330

Fattore M, Bruggemann R, Owsiński J (2011a) Using poset theory to compare fuzzy multidimensional material deprivation across regions. In: New perspectives in statistical modeling and data analysis. Springer, Berlin, pp 49–56

Fattore M, Maggino F, Greselin F (2011b) Socio-economic evaluation with ordinal variables: Integrating counting and poset approaches. Statistica & Applicazioni 1(2011):31–42

Fattore M, Maggino F, Colombo E (2012) From composite indicators to partial orders: evaluating socio-economic phenomena through ordinal data. In: Quality of life in Italy. Springer, Berlin, pp 41–68

R Core Team (2015) R: a language and environment for statistical computing. R Foundation for Statistical Computing, Vienna, Austria, 2015