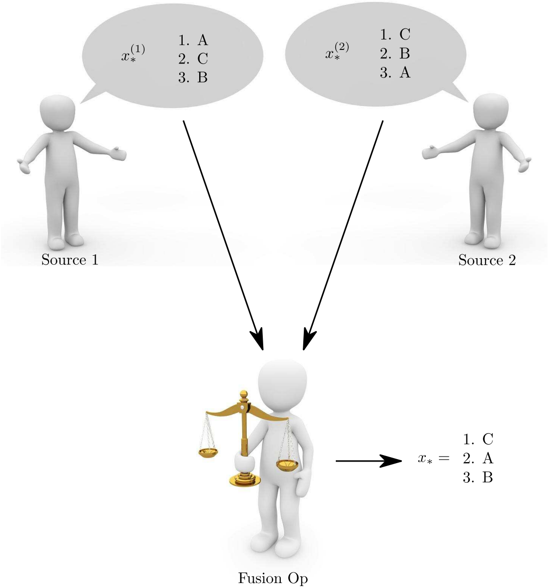
Chapter 4 : Ranked Data Fusion

John Klein
https://john-klein.github.io

Lille1 Université - CRIStAL UMR CNRS 9189

Ranked data fusion setting :



- Each advocacy $x^{(i)}$ is a set of possible values for x which are ranked from most preferred to least preferred.

- Let us give the following definition for ranked data fusion problems :

Definition

Ranked data fusion is a subclass of data fusion where advocacies live in $\mathbb{X} = \mathrm{perm}\left\{1, ..., |\mathcal{X}|\right\}$.

- In this chapter, we present two tallying algorithms for exhaustive ranked ballots.

- Extension are possible for settings where votes for a subset of $\mathcal{X}$ are allowed.

- Each advocacy $x^{(i)}$ is a set of possible values for x which are ranked from most preferred to least preferred.

- Let us give the following definition for ranked data fusion problems :

Definition

Ranked data fusion is a subclass of data fusion where advocacies live in $\mathbb{X} = \text{perm}\{1, .., |\mathcal{X}|\}$.

- In this chapter, we present two tallying algorithms for exhaustive ranked ballots.

- Extension are possible for settings where votes for a subset of $\mathcal{X}$ are allowed.

- Each advocacy $x^{(i)}$ is a set of possible values for x which are ranked from most preferred to least preferred.
- Let us give the following definition for ranked data fusion problems :

Definition

Ranked data fusion is a subclass of data fusion where advocacies live in $\mathbb{X} = \mathrm{perm}\left\{1, .., |\mathcal{X}|\right\}$.

- In this chapter, we present two tallying algorithms for exhaustive ranked ballots.
- Extension are possible for settings where votes for a subset of $\mathcal{X}$ are allowed.

- Each advocacy $x^{(i)}$ is a set of possible values for x which are ranked from most preferred to least preferred.

- Let us give the following definition for ranked data fusion problems :

Definition

Ranked data fusion is a subclass of data fusion where advocacies live in $\mathbb{X} = \text{perm}\left\{1, .., |\mathcal{X}|\right\}$.

- In this chapter, we present two tallying algorithms for exhaustive ranked ballots.

- Extension are possible for settings where votes for a subset of $\mathcal{X}$ are allowed.

- Each advocacy $x^{(i)}$ is a set of possible values for x which are ranked from most preferred to least preferred.

- Let us give the following definition for ranked data fusion problems :

Definition

Ranked data fusion is a subclass of data fusion where advocacies live in $\mathbb{X} = \mathrm{perm}\left\{1, .., |\mathcal{X}|\right\}$.

- In this chapter, we present two tallying algorithms for exhaustive ranked ballots.

- Extension are possible for settings where votes for a subset of $\mathcal{X}$ are allowed.

Instant Runoff Operator (IRO) :

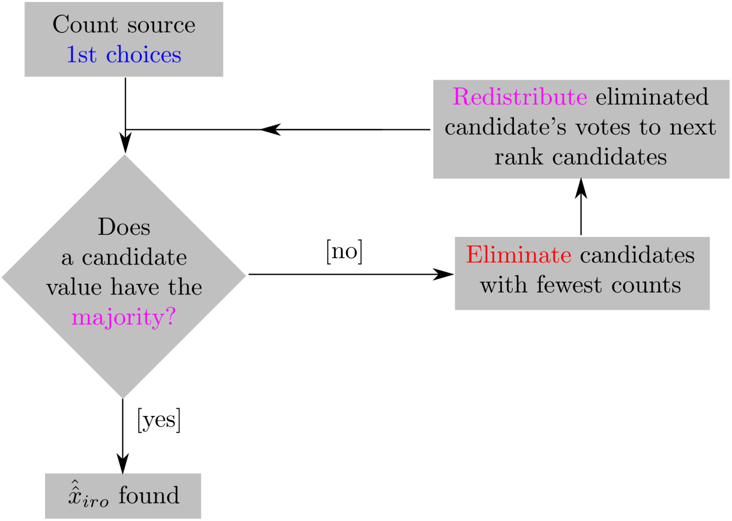


FIGURE – Flowchart of Instant Runoff Operator.

Instant Runoff Operator (IRO) : Notations

- Each advocacy $\mathbf{x}_*^{(i)}$ is a list.
- The j^{th} entry $x_{*j}^{(i)}$ of $\mathbf{x}_*^{(i)}$ is the value which was assigned rank j by the i^{th} source.
- $\mathbf{h}$: histogram computed from 1^{st} ranked values : $\left\{x_{*1}^{(i)}\right\}_{i=1}^{N_s}$.
  h_k : occurrences of 1^{st} ranks of the value $x_k \in \mathcal{X}$ [a].
- $j^{(i)}$: list index of the value currently supported by the i^{th} source.
- $k^{(i)}$: index in $\mathcal{X}$ of the value currently supported by the i^{th} source.

a. The index k does not depend on any underlying ordering relation on $\mathcal{X}$.

Instant Runoff Operator (IRO) : Notations

- Each advocacy $\mathbf{x}_*^{(i)}$ is a list.
- The j^{th} entry $x_{*j}^{(i)}$ of $\mathbf{x}_*^{(i)}$ is the value which was assigned rank j by the i^{th} source.
- $\mathbf{h}$: histogram computed from 1^{st} ranked values : $\left\{ x_{*1}^{(i)} \right\}_{i=1}^{N_s}$.
  h_k : occurrences of 1^{st} ranks of the value $x_k \in \mathcal{X}$ [a].
- $j^{(i)}$: list index of the value currently supported by the i^{th} source.
- $k^{(i)}$: index in $\mathcal{X}$ of the value currently supported by the i^{th} source.

---

a. The index k does not depend on any underlying ordering relation on $\mathcal{X}$.

Instant Runoff Operator (IRO) : Notations

- Each advocacy $\mathbf{x}_*^{(i)}$ is a list.
- The j^{th} entry $x_{*j}^{(i)}$ of $\mathbf{x}_*^{(i)}$ is the value which was assigned rank j by the i^{th} source.
- $\mathbf{h}$: histogram computed from 1st ranked values : $\left\{ x_{*1}^{(i)} \right\}_{i=1}^{N_s}$.
  h_k : occurrences of 1st ranks of the value $x_k \in \mathcal{X}$ [a].
- $j^{(i)}$: list index of the value currently supported by the i^{th} source.
- $k^{(i)}$: index in $\mathcal{X}$ of the value currently supported by the i^{th} source.

a. The index k does not depend on any underlying ordering relation on $\mathcal{X}$.

Instant Runoff Operator (IRO) : Notations

- Each advocacy $\mathbf{x}_*^{(i)}$ is a list.
- The j^{th} entry $x_{*j}^{(i)}$ of $\mathbf{x}_*^{(i)}$ is the value which was assigned rank j by the i^{th} source.
- $\mathbf{h}$: histogram computed from 1st ranked values : $\left\{ x_{*1}^{(i)} \right\}_{i=1}^{N_s}$.
  h_k : occurrences of 1st ranks of the value $x_k \in \mathcal{X}$ [a].
- $j^{(i)}$: list index of the value currently supported by the i^{th} source.
- $k^{(i)}$: index in $\mathcal{X}$ of the value currently supported by the i^{th} source.

a. The index k does not depend on any underlying ordering relation on $\mathcal{X}$.

Instant Runoff Operator (IRO) : Notations

- Each advocacy $\mathbf{x}_*^{(i)}$ is a list.
- The j^{th} entry $x_{*j}^{(i)}$ of $\mathbf{x}_*^{(i)}$ is the value which was assigned rank j by the i^{th} source.
- $\mathbf{h}$: histogram computed from 1^{st} ranked values : $\left\{ x_{*1}^{(i)} \right\}_{i=1}^{N_s}$.
  h_k : occurrences of 1^{st} ranks of the value $x_k \in \mathcal{X}$ [a].
- $j^{(i)}$: list index of the value currently supported by the i^{th} source.
- $k^{(i)}$: index in $\mathcal{X}$ of the value currently supported by the i^{th} source.

a. The index k does not depend on any underlying ordering relation on $\mathcal{X}$.

Instant Runoff Operator (IRO)

entries : $\left\{\mathbf{x}_*^{(i)}\right\}_{i=1}^{N_s}$, N_s,$\mathbf{h}$,$\ell = |\mathcal{X}|$ - Initialization :$\forall i, j^{(i)} \leftarrow 1$.

while $\max\limits_{1 \leq i \leq \ell} \{h_i\} < \frac{N_s}{2}$ **do**

Find the loser indice(s) : $L \leftarrow \arg\min\limits_{1 \leq k \leq \ell \text{ s.t. } h_k > 0} h_k$.

for i from 1 to N_s **do**

if $k^{(i)} \in L$ **then**

Move to next prefered value for source i : $j^{(i)} \leftarrow j^{(i)} + 1$.

Update histogram for remaining candidates : $h_{k^{(i)}} \leftarrow h_{k^{(i)}} + 1$.

end if

end for

for each k in L **do**

Update histogram for eliminated candidates : $h_k \leftarrow 0$.

end for

end while

Return winner : $x_* \leftarrow \arg\max\limits_{1 \leq i \leq \ell} \{h_i\}$

Example

Fusion of recommendation systems

- Suppose we trained N_S recommendation systems each returning a list of $|\mathcal{X}|=4$ objects.
- Each list is ordered from most preferred to least preferred.
- Let us also assume that they delivered only 4 different kinds of ballots.
- The table in the next slide gives the advocacies along with their occurences.
- Using the majority operator, site web A would win.

Ballot table for example

occurrences	42	26	15	17
ballot	A	B	C	D
	B	C	D	C
	C	D	B	B
	D	A	A	A

Pairwise Rank Operator (PRO) :

- Each advocacy $\mathbf{x}_*^{(i)}$ is a list.
- Reversed Logic : the j^{th} entry $x_{*j}^{(i)}$ of $\mathbf{x}_*^{(i)}$ is the rank assigned by the i^{th} source to the value j^{th} element of $\mathcal{X}$ [b].
- Note that algorithm PRO delivers no result if all wins are obtained with identical counts.

b. The index j of elements in $\mathcal{X}$ does not depend on any underlying ordering relation.

Pairwise Rank Operator (PRO) :

- Each advocacy $\mathbf{x}_*^{(i)}$ is a list.
- **Reversed Logic** : the j^{th} entry $x_{*j}^{(i)}$ of $\mathbf{x}_*^{(i)}$ is the rank assigned by the i^{th} source to the value j^{th} element of $\mathcal{X}$ [b].
- Note that algorithm PRO delivers no result if all wins are obtained with identical counts.

b. The index j of elements in $\mathcal{X}$ does not depend on any underlying ordering relation.

Pairwise Rank Operator (PRO) :

- Each advocacy $\mathbf{x}_*^{(i)}$ is a list.

- **Reversed Logic** : the j^{th} entry $x_{*j}^{(i)}$ of $\mathbf{x}_*^{(i)}$ is the rank assigned by the i^{th} source to the value j^{th} element of $\mathcal{X}$ [b].

- Note that algorithm PRO delivers no result if all wins are obtained with identical counts.

b. The index j of elements in $\mathcal{X}$ does not depend on any underlying ordering relation.

Pairwise Rank Operator (PRO) : step1 [Preference counting]

entries : $\left\{\mathbf{x}_*^{(i)}\right\}_{i=1}^{N_s}$, N_s, $\ell = |\mathcal{X}|$.
Initialize $\mathbf{M}$ and $\mathbf{W}$ are $\ell \times \ell$ as null matrices.
for i from 1 to N_s **do**
  for $1 \leq j < j' \leq \ell$ **do**
    if $x_{*j}^{(i)} < x_{*j'}^{(i)}$ **then**
      Add one count to j over j' : $M_{jj'} \leftarrow M_{jj'} + 1$.
    else
      Add one count to j' over j : $M_{j'j} \leftarrow M_{j'j} + 1$.
    end if
  end for
end for
$\mathbf{M} \leftarrow \frac{\mathbf{M}}{N_s}$.

Pairwise Rank Operator (PRO) : step2 [Victory counting]

...

for j from 1 to ℓ **do**

  for j' from 1 to ℓ **do**

    if $M_{jj'} > \frac{1}{2}$ **then**

      Count one win for j over j' : $W_{jj'} \leftarrow 1$.

    end if

  end for

end for

Pairwise Rank Operator (PRO) : step3 [Global winner search]

...

Find the set of candidates with highest win counts :

$A \leftarrow \underset{1 \leq j \leq \ell}{\arg\max} \left\{ \sum_{j'=1}^{\ell} W_{jj'} \right\}$.

while $|A| > 1$ **do**

Find the index pair $(u; v)$ corresponding to the weakest win :

$(u; v) \leftarrow \underset{1 \leq j, j' \leq \ell \,|\, W_{jj'}=1}{\arg\min} \left\{ M_{jj'} \right\}$.

Erase this win : $W_{uv} \leftarrow 0$.

Update $A \leftarrow \underset{1 \leq j \leq \ell}{\arg\max} \left\{ \sum_{j'=1}^{\ell} W_{jj'} \right\}$.

end while

Return winner : $x_* \leftarrow A$

Example (continued)

Fusion of recommendation systems

- Suppose we trained N_S recommendation systems each returning a list of $|\mathcal{X}|$=4 objects.
- Each list is ordered from most preferred to least preferred.
- Let us also assume that they delivered only 4 different kinds of ballots.
- The table in the next slide gives the advocacies along with their occurrences.

Table for example

occurrences	42	26	15	17
ballot	A	B	C	D
	B	C	D	C
	C	D	B	B
	D	A	A	A

Ranked Data Fusion : concluding remarks

- There are many other voting systems for ranked ballots in the literature.
- Each of them are often justified with respect to a given criterion on the quality of the result.
- For instance, one of the criterion that motivated the design of $\hat{\hat{x}}_{pro}$ is the Condorcet criterion which reads :

« *The winner must defeat any other candidate in pairwise elections.* »