Computational Approach of Musical Orchestration

Constrained Multiobjective Optimization of Sound Combinations in Large Instrument Sample Databases

December 16th, 2008

Grégoire Carpentier - Ph.D. Defense
IRCAM - Music Representation Group

Supervisors : Gérard Assayag (IRCAM) & Emmanuel Saint-James (LIP6)

Contents

Contents

Contents

Contents

Contents

Contents

Contents

Contents

Computer-Aided Composition

Computer-Aided Composition

- Formalizing of musical structures

Computer-Aided Composition

- Formalizing of musical structures
- Formalizing processes

Computer-Aided Composition

- Formalizing of musical structures
- Formalizing processes

- Tackling different aspects of musical writing:

Computer-Aided Composition

- Formalizing of musical structures
- Formalizing processes

- Tackling different aspects of musical writing:
  - ☑ Durations, rythm

Computer-Aided Composition

- Formalizing of musical structures
- Formalizing processes

- Tackling different aspects of musical writing:
  - ☑ Durations, rythm
  - ☑ Pitches, melody, harmony

Computer-Aided Composition

- Formalizing of musical structures
- Formalizing processes

- Tackling different aspects of musical writing:
  - ☑ Durations, rythm
  - ☑ Pitches, melody, harmony
  - ☑ Time representations

Computer-Aided Composition

- Formalizing of musical structures
- Formalizing processes

- Tackling different aspects of musical writing:
  - ☑ Durations, rythm
  - ☑ Pitches, melody, harmony
  - ☑ Time representations
  - ☑ Spatialization

Computer-Aided Composition

- Formalizing of musical structures
- Formalizing processes

- Tackling different aspects of musical writing:
  - ☑ Durations, rythm
  - ☑ Pitches, melody, harmony
  - ☑ Time representations
  - ☑ Spatialization
  - ☑ Sound synthesis

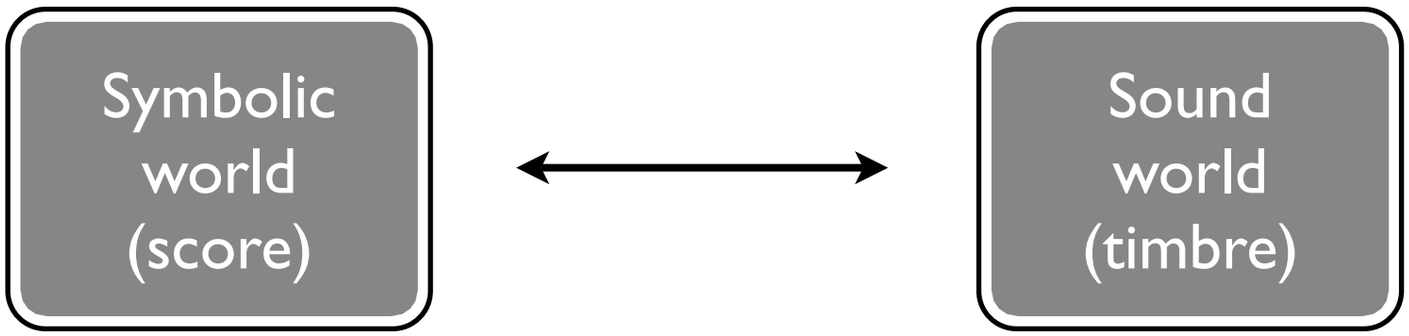
Computer-Aided Composition

- Formalizing of musical structures
- Formalizing processes

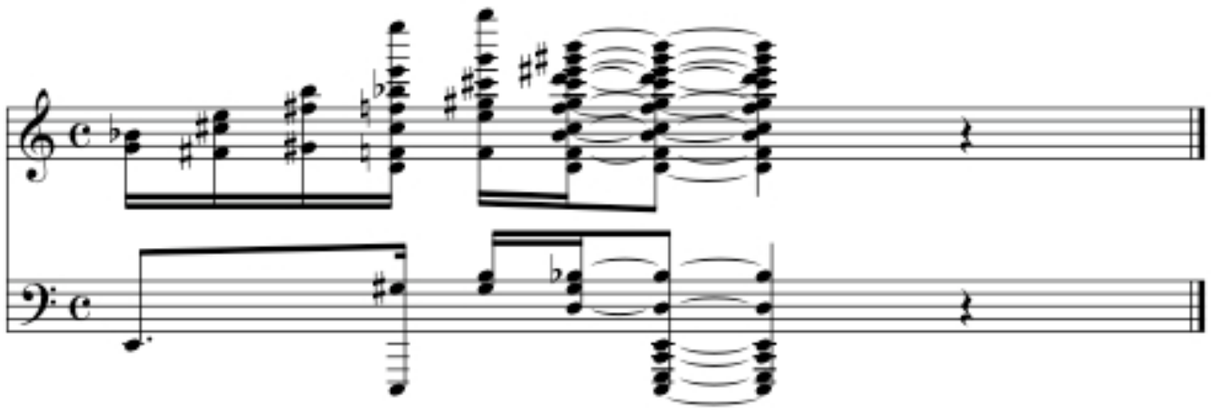
- Tackling different aspects of musical writing:
  - ☑ Durations, rythm
  - ☑ Pitches, melody, harmony
  - ☑ Time representations
  - ☑ Spatialization
  - ☑ Sound synthesis

  - ☐ **Instrumental timbre ?**

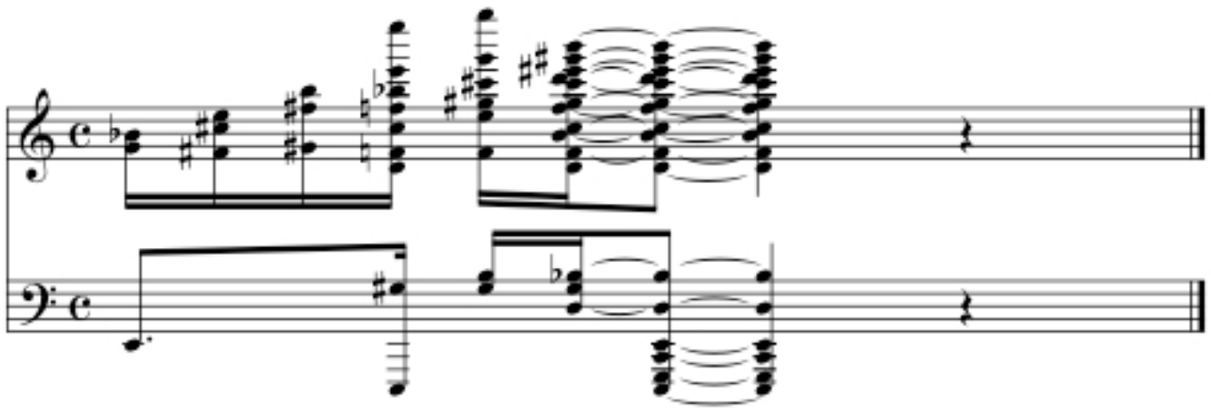
Orchestration - timbre



Orchestration : projection



Orchestration : projection

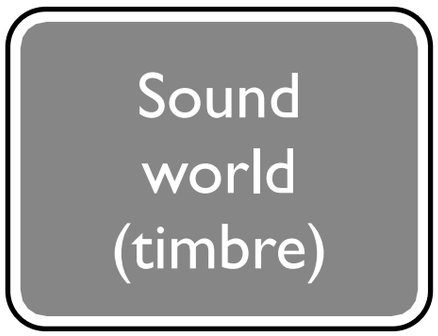


Projection

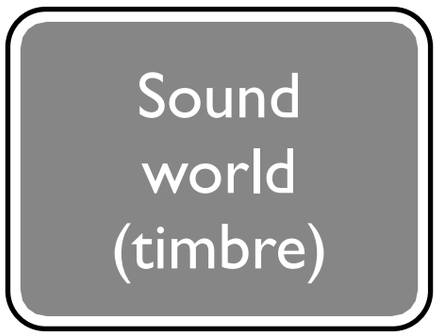
Sound world (timbre)

Orchestration : induction

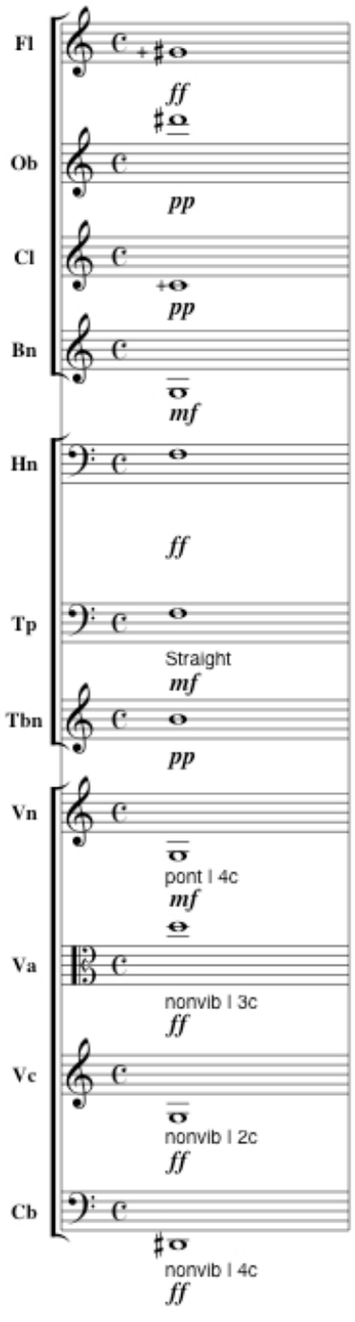
Orchestration : induction



Orchestration : induction



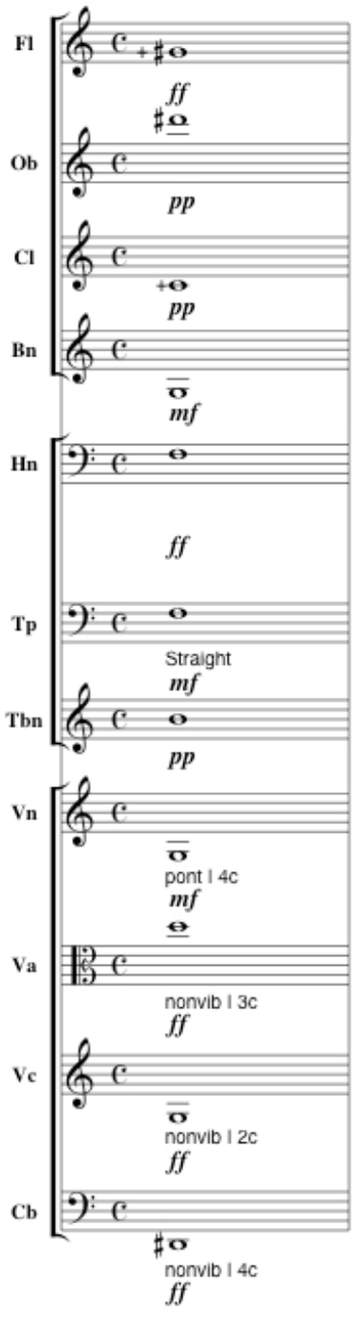
Orchestration : induction



Induction

Sound world (timbre)

Orchestration : induction



Induction

Sound world (timbre)

Orchestration in practice

Orchestration in practice

- Personal knowledge

Orchestration in practice

- Personal knowledge → Necessarily restricted

Orchestration in practice

- Personal knowledge → Necessarily restricted
- Orchestration Treatises
  - [Berlioz1855]
  - [Koechlin1943]
  - [Piston1955]
  - [Rimski-Korsakov1912†]

Orchestration in practice

- Personal knowledge → Necessarily restricted
- Orchestration Treatises → Obviously outdated
  - [Berlioz1855]
  - [Koechlin1943]
  - [Piston1955]
  - [Rimski-Korsakov1912†]

Orchestration in practice

- Personal knowledge → Necessarily restricted
- Orchestration Treatises → Obviously outdated
  - [Berlioz1855]
  - [Koechlin1943]
  - [Piston1955]
  - [Rimski-Korsakov1912†]
- Current computer orchestration tools
  - [Psenicka2003]
  - [Rose&Hetrick2005]
  - [Hummel2005]

Orchestration in practice

- Personal knowledge → Necessarily restricted
- Orchestration Treatises → Obviously outdated
  - [Berlioz1855]
  - [Koechlin1943]
  - [Piston1955]
  - [Rimski-Korsakov1912†]
- Current computer orchestration tools → Monoobjective approach of timbre
  - [Psenicka2003]
  - [Rose&Hetrick2005]
  - [Hummel2005]

Orchestration in practice

- Personal knowledge → Necessarily restricted
- Orchestration Treatises → Obviously outdated
  - [Berlioz1855]
  - [Koechlin1943]
  - [Piston1955]
  - [Rimski-Korsakov1912†]
- Current computer orchestration tools → Monoobjective approach of timbre; Do not handle combinatorial issues
  - [Psenicka2003]
  - [Rose&Hetrick2005]
  - [Hummel2005]

Contents

Problem statement

Problem statement

How can I use an orchestra to reproduce a timbre target within a given compositional context?

Problem statement

How can I use an orchestra to reproduce a timbre target within a given compositional context?

How can I find an instrument sound combination that:

Problem statement

How can I use an orchestra to reproduce a timbre target within a given compositional context?

How can I find an instrument sound combination that:

- Best matches a given target sound?

Problem statement

How can I use an orchestra to reproduce a timbre target within a given compositional context?

How can I find an instrument sound combination that:

- Best matches a given target sound?
- Fits writing constraints specified by the composer?

Problem statement

How can I use an orchestra to reproduce a timbre target within a given compositional context?

How can I find an instrument sound combination that:

- Best matches a given target sound?
- Fits writing constraints specified by the composer?

In computer terms:

Problem statement

How can I use an orchestra to reproduce a timbre target within a given compositional context?

How can I find an instrument sound combination that:

- Best matches a given target sound?
- Fits writing constraints specified by the composer?

In computer terms:

- A combinatorial optimization problem defined on a timbre description scheme

Problem statement

How can I use an orchestra to reproduce a timbre target within a given compositional context?

How can I find an instrument sound combination that:

- Best matches a given target sound?
- Fits writing constraints specified by the composer?

In computer terms:

- A combinatorial optimization problem defined on a timbre description scheme
- A constraint solving problem on the variables of musical writing

Strategy

Strategy

- Combinatorial optimization problem: the ***orchidée*** algorithm

Strategy

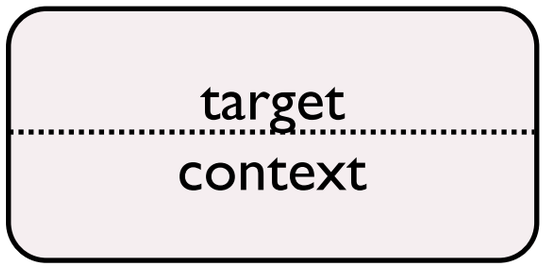
- Combinatorial optimization problem: the ***orchidée*** algorithm
- Constraint solving problem: the ***cdcsolver*** algorithm

Strategy

- Combinatorial optimization problem: the ***orchidée*** algorithm
- Constraint solving problem: the ***cdcsolver*** algorithm
- Collaboration between the two methods

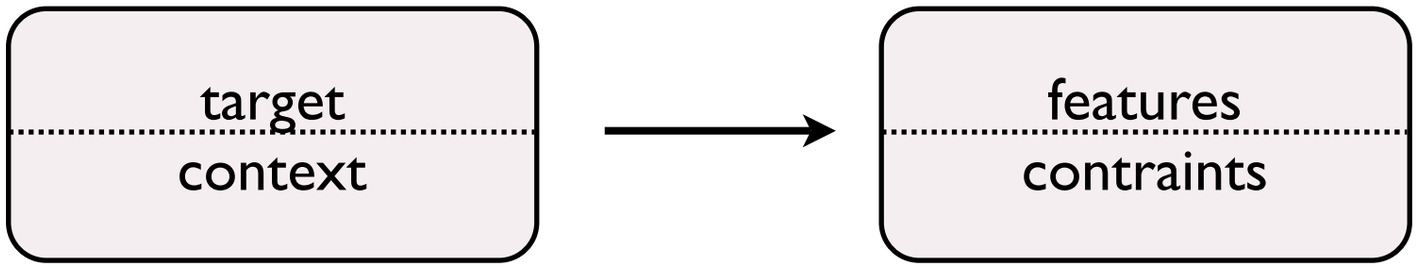
Strategy

- Combinatorial optimization problem: the ***orchidée*** algorithm
- Constraint solving problem: the ***cdcsolver*** algorithm
- Collaboration between the two methods



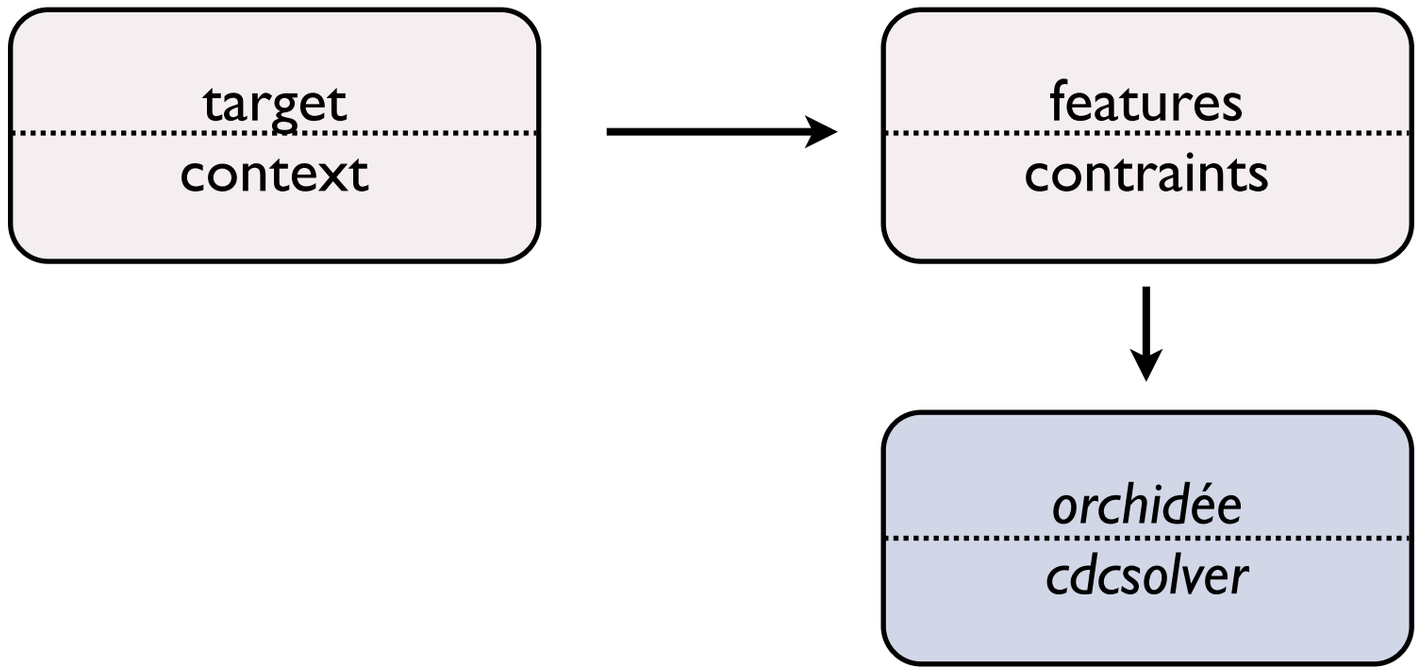
Strategy

- Combinatorial optimization problem: the ***orchidée*** algorithm
- Constraint solving problem: the ***cdcsolver*** algorithm
- Collaboration between the two methods



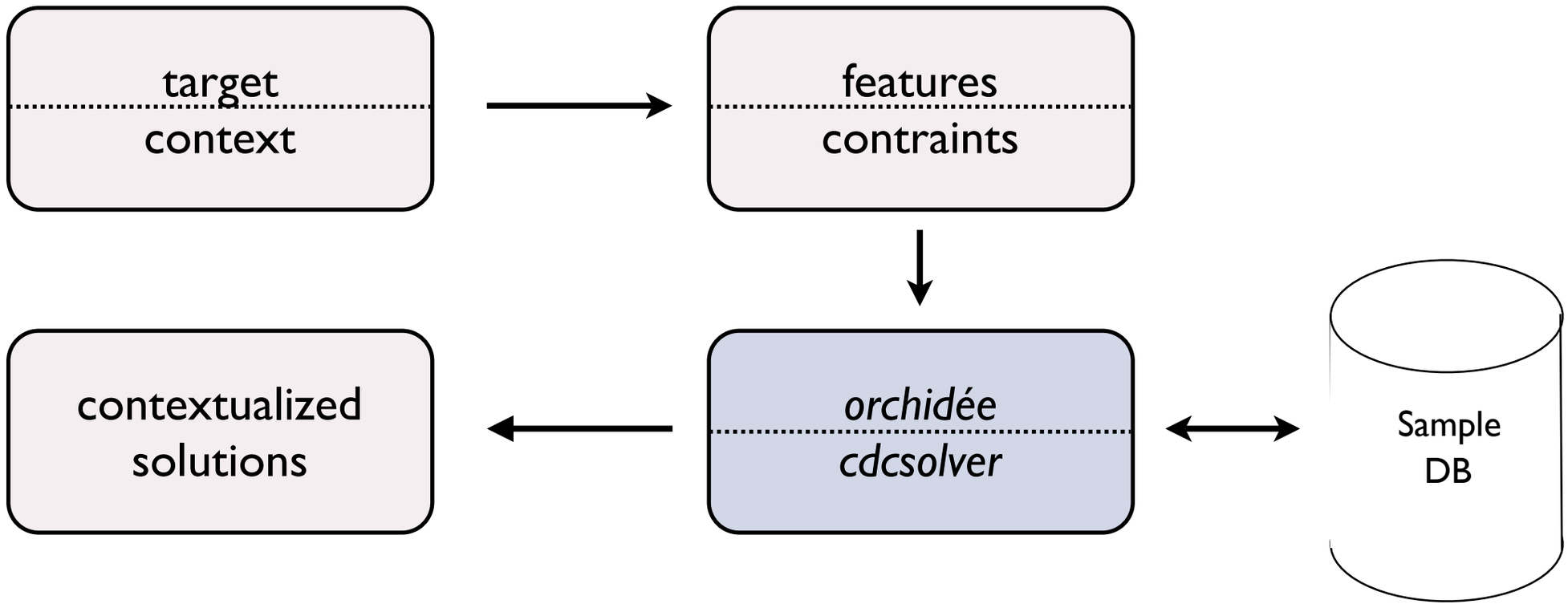
Strategy

- Combinatorial optimization problem: the ***orchidée*** algorithm
- Constraint solving problem: the ***cdcsolver*** algorithm
- Collaboration between the two methods



Strategy

- Combinatorial optimization problem: the ***orchidée*** algorithm
- Constraint solving problem: the ***cdcsolver*** algorithm
- Collaboration between the two methods



Contents

Timbre similarity (1)

Timbre similarity (1)

- Timbre perception is multidimensional

Timbre similarity (1)

- Timbre perception is multidimensional

Timbre similarity (1)

- Timbre perception is multidimensional
- Correlation between perceptual dimensions and sound features [McAdams+1995] [Peeters2004]:

Timbre similarity (1)

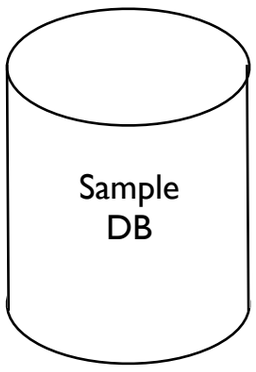
- Timbre perception is multidimensional
- Correlation between perceptual dimensions and sound features [McAdams+1995] [Peeters2004]:
    - Spectral centroid <=> brightness

Timbre similarity (1)

- Timbre perception is multidimensional
- Correlation between perceptual dimensions and sound features [McAdams+1995] [Peeters2004]:
    - Spectral centroid <=> brightness
    - Attack time <=> percussive / sustained sounds

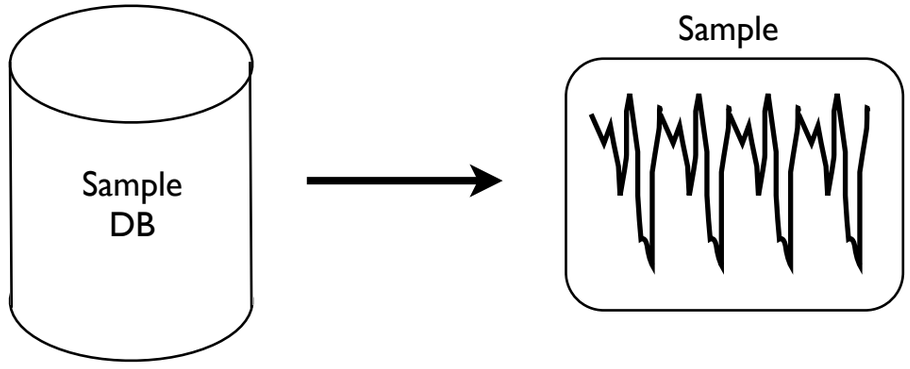
Timbre similarity (1)

- Timbre perception is multidimensional
- Correlation between perceptual dimensions and sound features [McAdams+1995] [Peeters2004]:
    - Spectral centroid <=> brightness
    - Attack time <=> percussive / sustained sounds



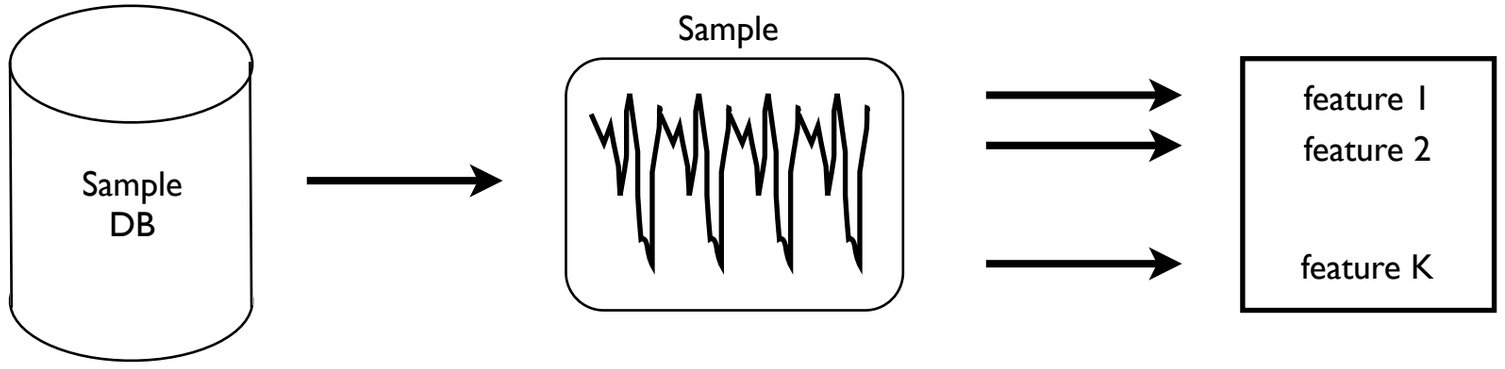
Timbre similarity (1)

- Timbre perception is multidimensional
- Correlation between perceptual dimensions and sound features [McAdams+1995] [Peeters2004]:
    - Spectral centroid <=> brightness
    - Attack time <=> percussive / sustained sounds



Timbre similarity (1)

- Timbre perception is multidimensional
- Correlation between perceptual dimensions and sound features [McAdams+1995] [Peeters2004]:
    - Spectral centroid <=> brightness
    - Attack time <=> percussive / sustained sounds

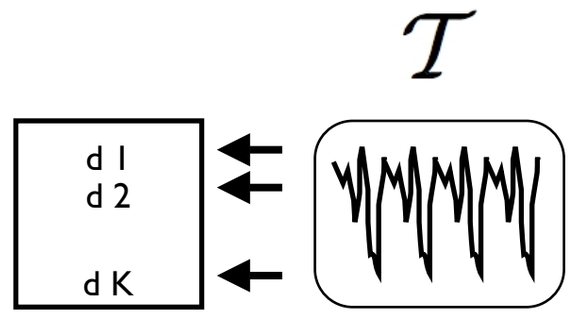


Timbre similarity (2)

- Hypothesis: Sound combination feature set can be predicted from the values of components features

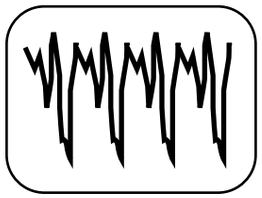
Timbre similarity (2)

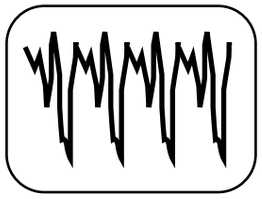
- Hypothesis: Sound combination feature set can be predicted from the values of components features

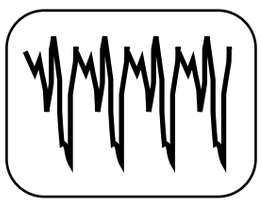


Timbre similarity (2)

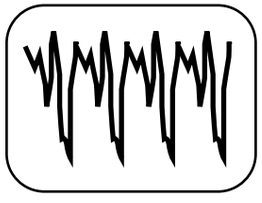
- Hypothesis: Sound combination feature set can be predicted from the values of components features



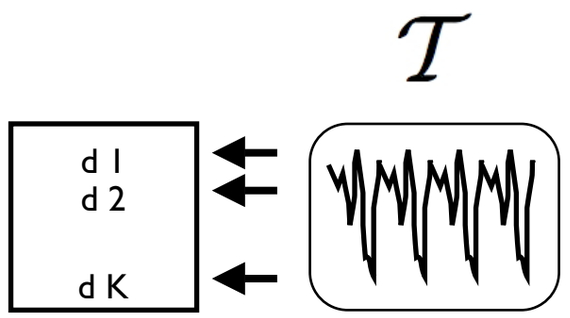




$\mathcal{S}$

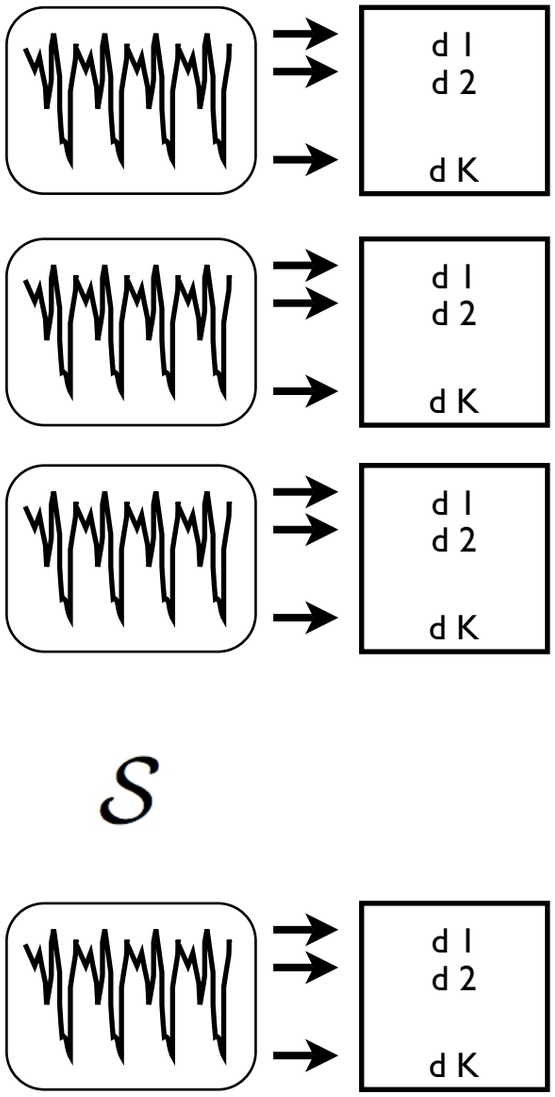


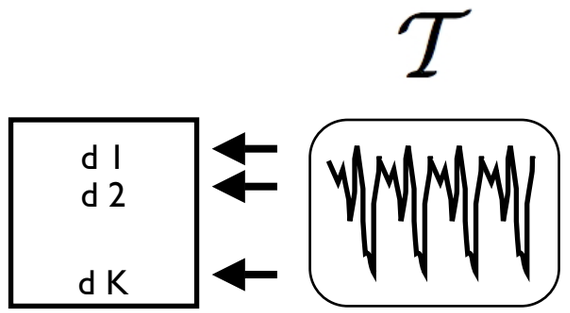
$\mathcal{T}$



Timbre similarity (2)

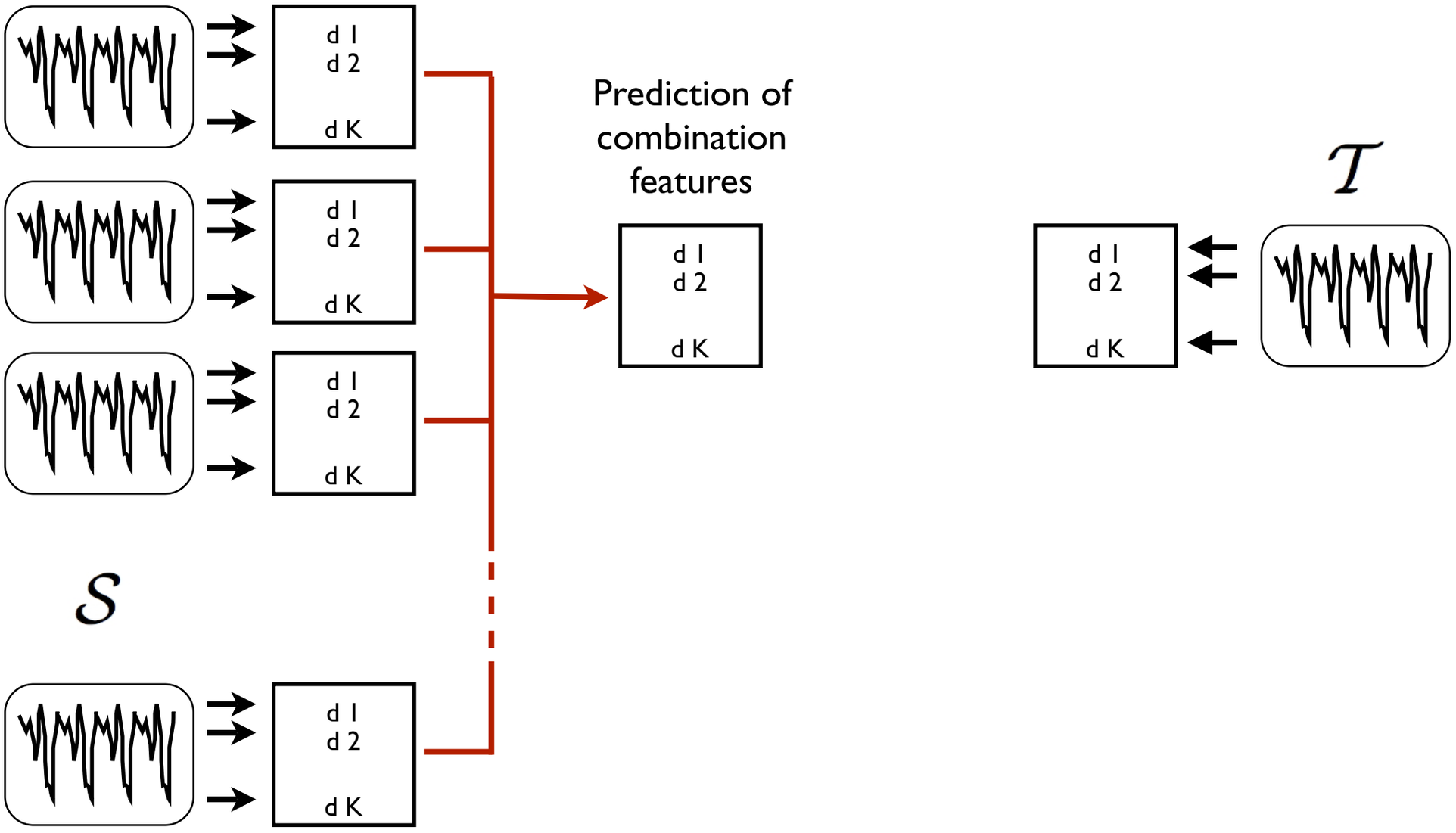
- Hypothesis: Sound combination feature set can be predicted from the values of components features





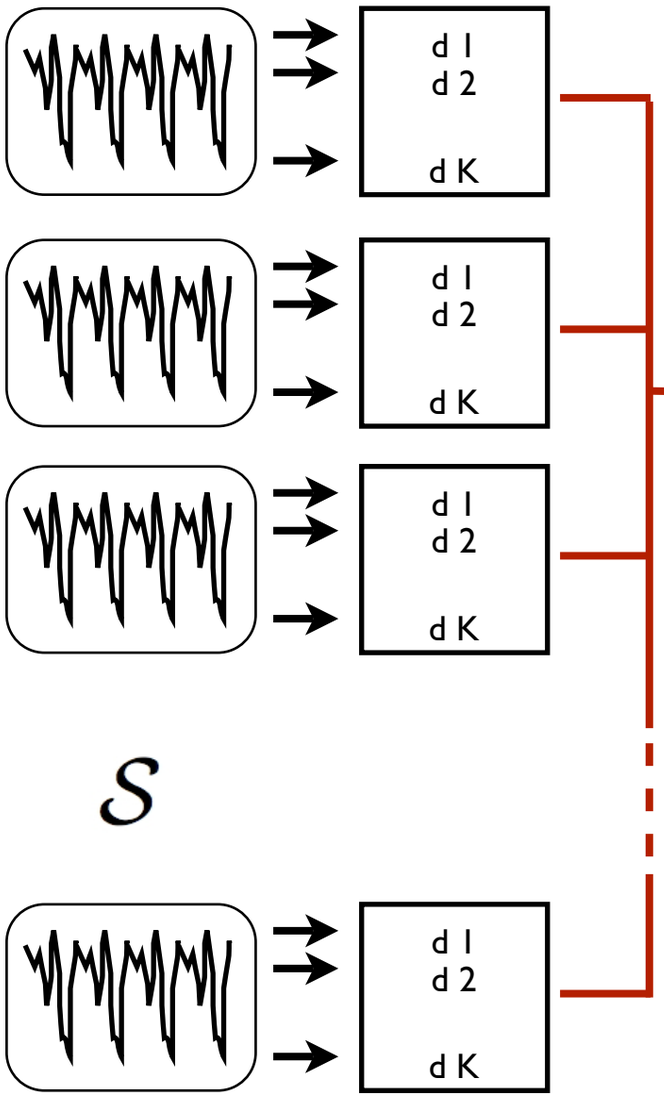
Timbre similarity (2)

- Hypothesis: Sound combination feature set can be predicted from the values of components features



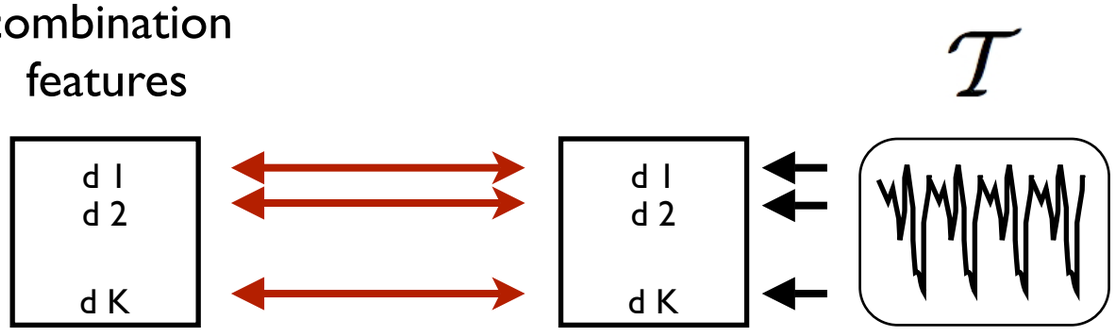
Timbre similarity (2)

- Hypothesis: Sound combination feature set can be predicted from the values of components features



$\mathcal{S}$

Prediction of combination features



$\mathcal{T}$

Dissimilarity measures along each perceptual dimension

$$\left\{ D_{\mathcal{T}}^{k}(\mathcal{S}) \mid k = 1, \ldots, K \right\}$$

Sound features

- Spectral centroid (brightness) [McAdams+1995]
- Spectral spread (volume) [Chiasson2007]
- Resolved partials (harmonic tone)

- Time and noise features are not considered

- Preliminary tasks:
  - Sound combinations features prediction functions
  - Perceptual dissimilarity functions $\left\{D_T^k(\mathcal{S}) \mid k = 1, \ldots, K\right\}$

Multiobjective approach

Multiobjective approach

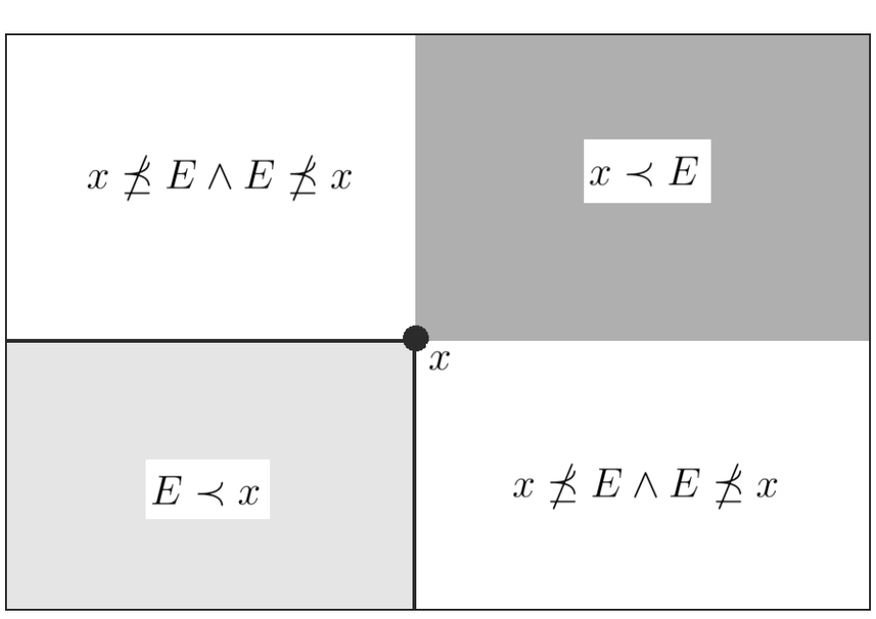
- Relative importance of perceptual dimensions cannot be known without prior information on listening preferences

Multiobjective approach

- Relative importance of perceptual dimensions cannot be known without prior information on listening preferences
- Multiobjective optimization: Jointly minimize $\left\{D_T^k(\mathcal{S}) \mid k = 1, \ldots, K\right\}$

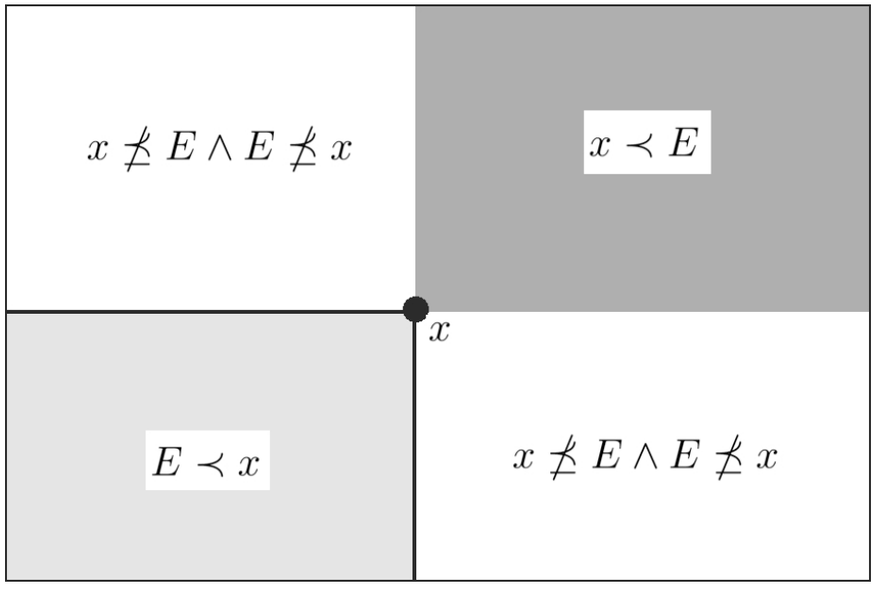
Multiobjective approach

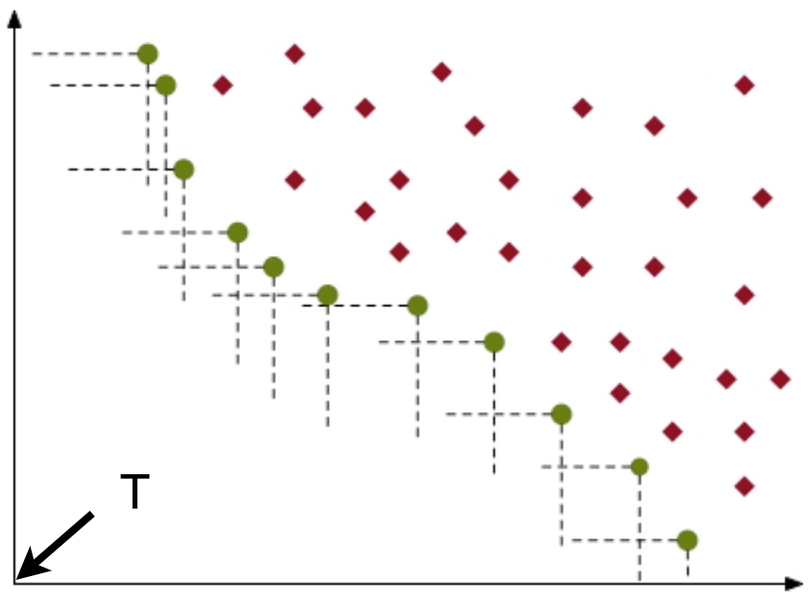
- Relative importance of perceptual dimensions cannot be known without prior information on listening preferences
- Multiobjective optimization: Jointly minimize $\left\{D_T^k(\mathcal{S}) \mid k = 1, \ldots, K\right\}$
- Pareto dominance: $x \preceq y \quad \Leftrightarrow \quad \forall k,\ x_k \leq y_k$



Multiobjective approach

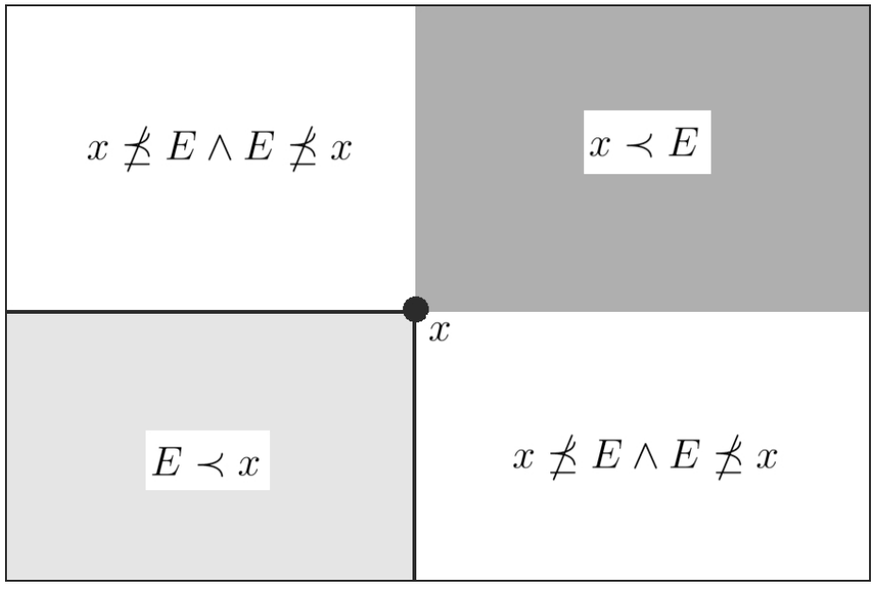
- Relative importance of perceptual dimensions cannot be known without prior information on listening preferences
- Multiobjective optimization: Jointly minimize $\left\{D_T^k(\mathcal{S}) \mid k = 1, \ldots, K\right\}$
- Pareto dominance: $\mathcal{S}_1 \preceq \mathcal{S}_2 \quad \Leftrightarrow \quad \forall k,\ D_T^k(\mathcal{S}_1) \leq D_T^k(\mathcal{S}_2)$

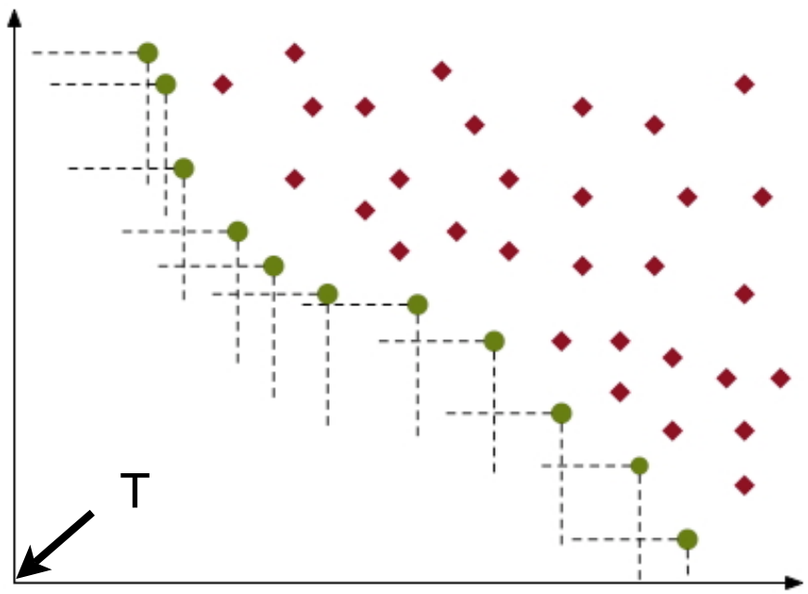




Multiobjective approach

- Relative importance of perceptual dimensions cannot be known without prior information on listening preferences
- Multiobjective optimization: Jointly minimize $\left\{D_T^k(\mathcal{S}) \mid k = 1, \ldots, K\right\}$
- Pareto dominance: $\mathcal{S}_1 \preceq \mathcal{S}_2 \quad \Leftrightarrow \quad \forall k,\ D_T^k(\mathcal{S}_1) \leq D_T^k(\mathcal{S}_2)$





- Set of optimal solutions (implicitly corresponding to different listening preferences)

Formalizing the optimization problem

Formalizing the optimization problem

- Orchestra composed of $\mathcal{I}$ instruments: $\mathcal{I}$ variables problem

Formalizing the optimization problem

- Orchestra composed of $\mathcal{I}$ instruments: $\mathcal{I}$ variables problem
- Domain of each variable: $\bar{E}_i = E_i \cup \{e\}$

Formalizing the optimization problem

- Orchestra composed of $\mathcal{I}$ instruments: $\mathcal{I}$ variables problem
- Domain of each variable: $\bar{E}_i = E_i \cup \{e\}$
- Problem:

$$\begin{cases} \min\limits_{\mathcal{S}} D_T^k(\mathcal{S}) = D_T^k(i_1, \ldots, i_{\mathcal{I}}) \ , \ k \in \{1, \ldots, K\} \\ \text{s.t. } \mathcal{S} \in \{\bar{E}_1 \times \cdots \times \bar{E}_{\mathcal{I}}\} \end{cases} \qquad (P)$$

A NP-hard problem

A NP-hard problem

Consider a 5000 sample sound database. Consider an 11 instruments orchestra in which only 4 can play simultaneously a given 4-note chord. Thus:

A NP-hard problem

Consider a 5000 sample sound database. Consider an 11 instruments orchestra in which only 4 can play simultaneously a given 4-note chord. Thus:

- There are around a billon a feasible combinations

A NP-hard problem

Consider a 5000 sample sound database. Consider an 11 instruments orchestra in which only 4 can play simultaneously a given 4-note chord. Thus:

- There are around a billon a feasible combinations
- Computing their features takes around 20 minutes with 3 perceptual dimensions

A NP-hard problem

Consider a 5000 sample sound database. Consider an 11 instruments orchestra in which only 4 can play simultaneously a given 4-note chord. Thus:

- There are around a billon a feasible combinations
- Computing their features takes around 20 minutes with 3 perceptual dimensions

Taking into account that:

A NP-hard problem

Consider a 5000 sample sound database. Consider an 11 instruments orchestra in which only 4 can play simultaneously a given 4-note chord. Thus:

- There are around a billon a feasible combinations
- Computing their features takes around 20 minutes with 3 perceptual dimensions

Taking into account that:

- A big orchestra may contain around a hundred instruments;

A NP-hard problem

Consider a 5000 sample sound database. Consider an 11 instruments orchestra in which only 4 can play simultaneously a given 4-note chord. Thus:

- There are around a billon a feasible combinations
- Computing their features takes around 20 minutes with 3 perceptual dimensions

Taking into account that:

- A big orchestra may contain around a hundred instruments;
- There might be around ten perceptual features;

A NP-hard problem

Consider a 5000 sample sound database. Consider an 11 instruments orchestra in which only 4 can play simultaneously a given 4-note chord. Thus:

- There are around a billon a feasible combinations
- Computing their features takes around 20 minutes with 3 perceptual dimensions

Taking into account that:

- A big orchestra may contain around a hundred instruments;
- There might be around ten perceptual features;
- Instrument sound databases may hold hundred of thousands items;

A NP-hard problem

Consider a 5000 sample sound database. Consider an 11 instruments orchestra in which only 4 can play simultaneously a given 4-note chord. Thus:

- There are around a billon a feasible combinations
- Computing their features takes around 20 minutes with 3 perceptual dimensions

Taking into account that:

- A big orchestra may contain around a hundred instruments;
- There might be around ten perceptual features;
- Instrument sound databases may hold hundred of thousands items;

Complete resolution methods cannot help here.

Metaheuristics are required.

GA optimization: the *orchidée* algorithm

GA optimization: the *orchidée* algorithm

- Evolutionary algorithm (using a population of individuals)

GA optimization: the *orchidée* algorithm

- Evolutionary algorithm (using a population of individuals)
- Each individual is represented by a set of genes (chromosome)

GA optimization: the *orchidée* algorithm

- Evolutionary algorithm (using a population of individuals)
- Each individual is represented by a set of genes (chromosome)

- ***Orchidée***: integer tuple encoding (“orchestra” representation)

GA optimization: the *orchidée* algorithm

- Evolutionary algorithm (using a population of individuals)
- Each individual is represented by a set of genes (chromosome)

- ***Orchidée***: integer tuple encoding (“orchestra” representation)

Vn1	art-harm B6 p
Vn2	
Alt	sul-ponticello D4 f
Vc	non-vibrato G3 mf

Vn1	art-harm B6 p
Vn2	sul-tasto Eb5+ mf
Alt	sul-ponticello D4 f
Vc	non-vibrato G3 f

Vn1	
Vn2	sul-tasto Eb5+ mf
Alt	sul-ponticello D4 f
Vc	non-vibrato G3 f

Vn1	art-harm B6 p
Vn2	sul-tasto Eb5+ mf
Alt	sul-ponticello D4 f
Vc	non-vibrato G3 mf

Vn1	non-vibrato B5+ p
Vn2	
Alt	sul-ponticello D4 f
Vc	non-vibrato G3 mf

Vn1	non-vibrato B5+ p
Vn2	
Alt	
Vc	non-vibrato G3 mf

GA optimization: the *orchidée* algorithm

- Evolutionary algorithm (using a population of individuals)
- Each individual is represented by a set of genes (chromosome)

- ***Orchidée***: integer tuple encoding (“orchestra” representation)

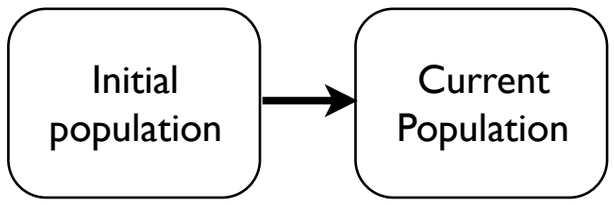
Vn1	art-harm B6 p	Vn1	art-harm B6 p	Vn1	
Vn2		Vn2	sul-tasto Eb5+ mf	Vn2	sul-tasto Eb5+ mf
Alt	sul-ponticello D4 f	Alt	sul-ponticello D4 f	Alt	sul-ponticello D4 f
Vc	non-vibrato G3 mf	Vc	non-vibrato G3 f	Vc	non-vibrato G3 f
Vn1	art-harm B6 p	Vn1	non-vibrato B5+ p	Vn1	non-vibrato B5+ p
Vn2	sul-tasto Eb5+ mf	Vn2		Vn2	
Alt	sul-ponticello D4 f	Alt	sul-ponticello D4 f	Alt	
Vc	non-vibrato G3 mf	Vc	non-vibrato G3 mf	Vc	non-vibrato G3 mf

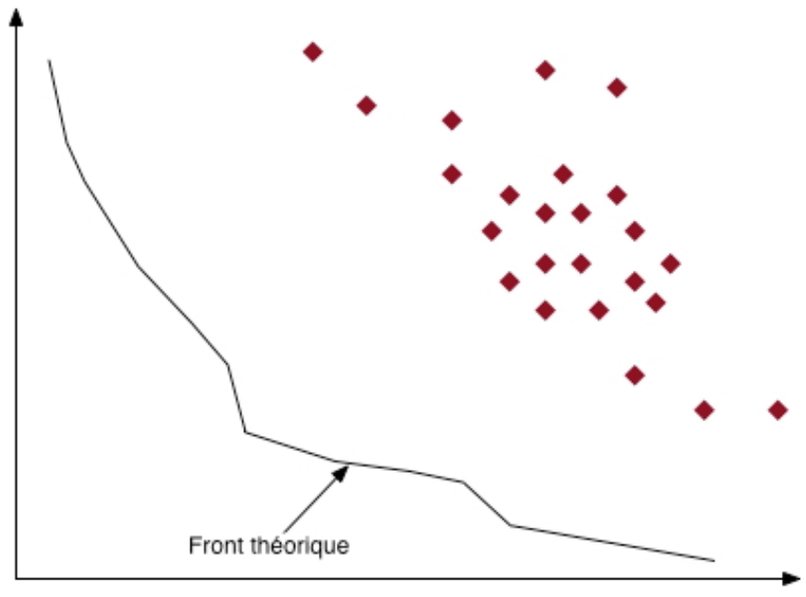
- ***Orchidée***: user preferences guessing mechanism

GA optimization: the *orchidée* algorithm

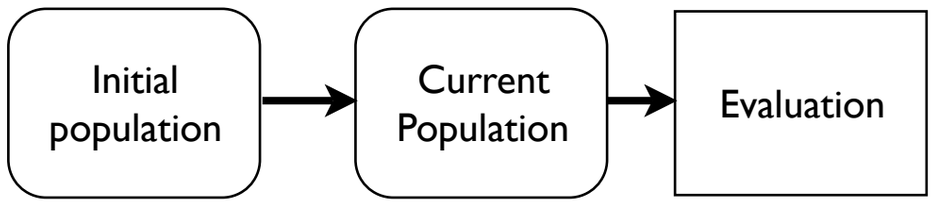
Initial population

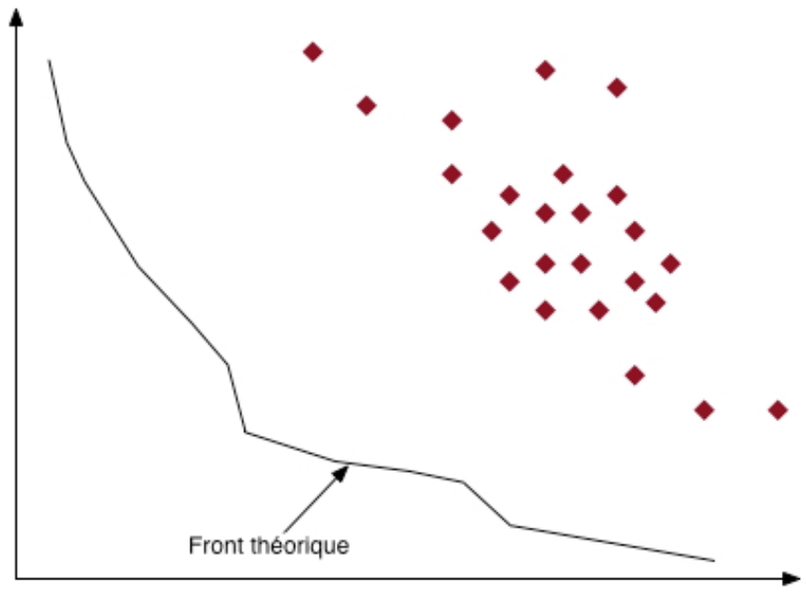
GA optimization: the *orchidée* algorithm



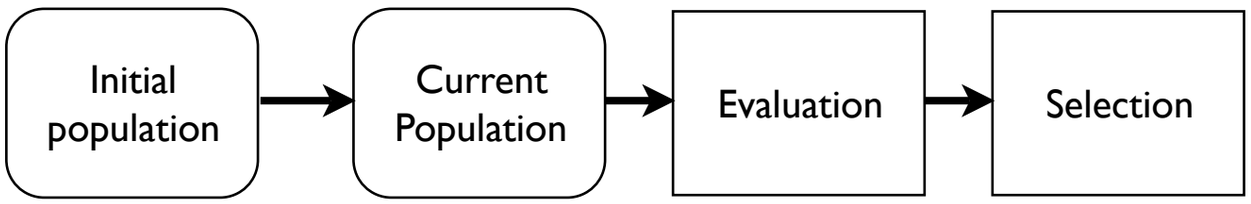


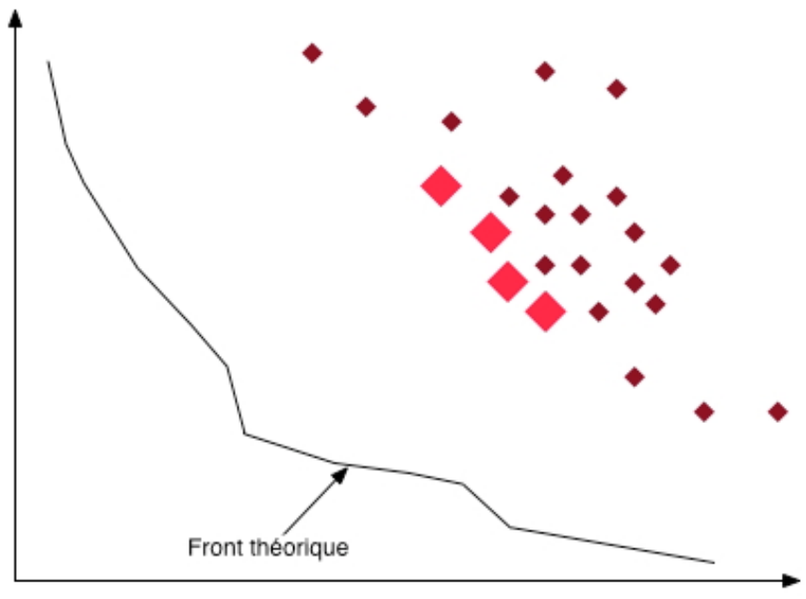
GA optimization: the *orchidée* algorithm



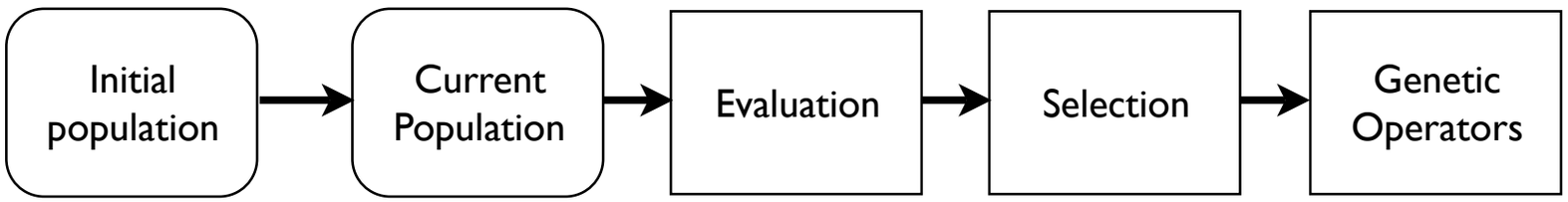


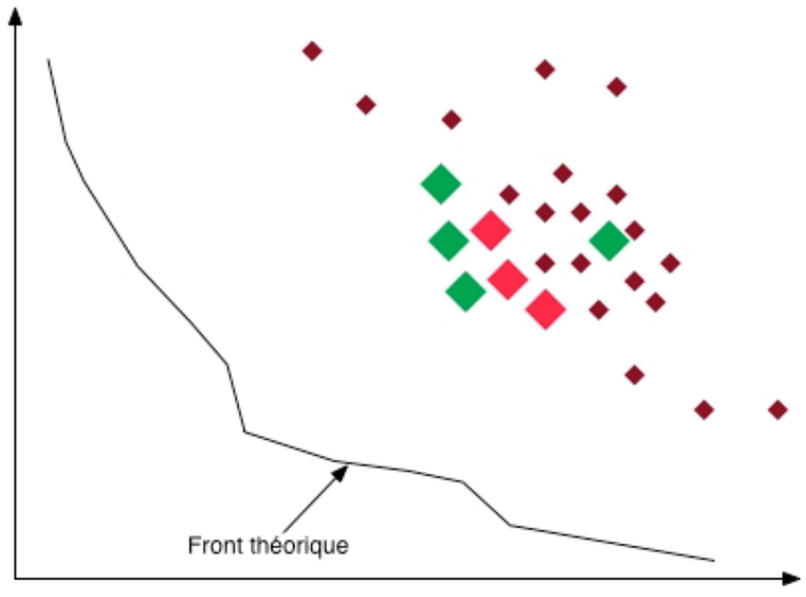
GA optimization: the *orchidée* algorithm



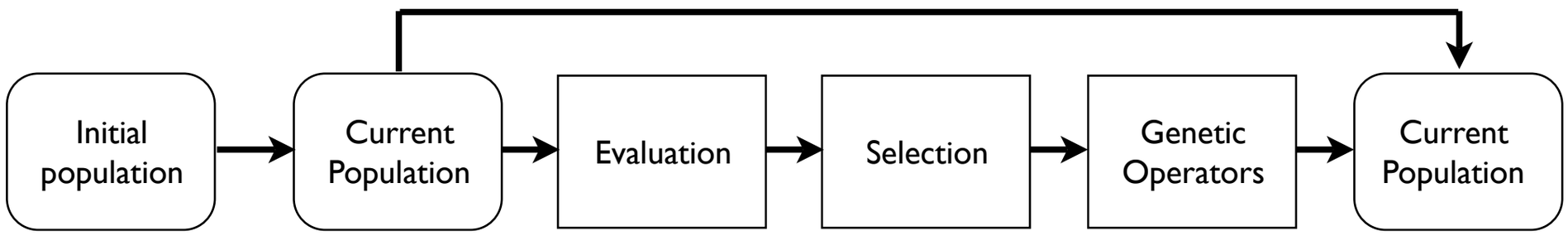


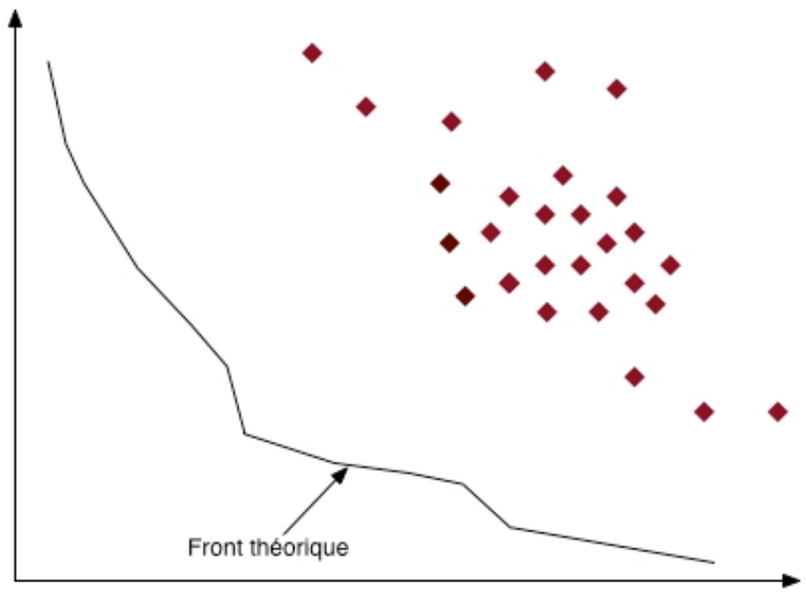
GA optimization: the *orchidée* algorithm



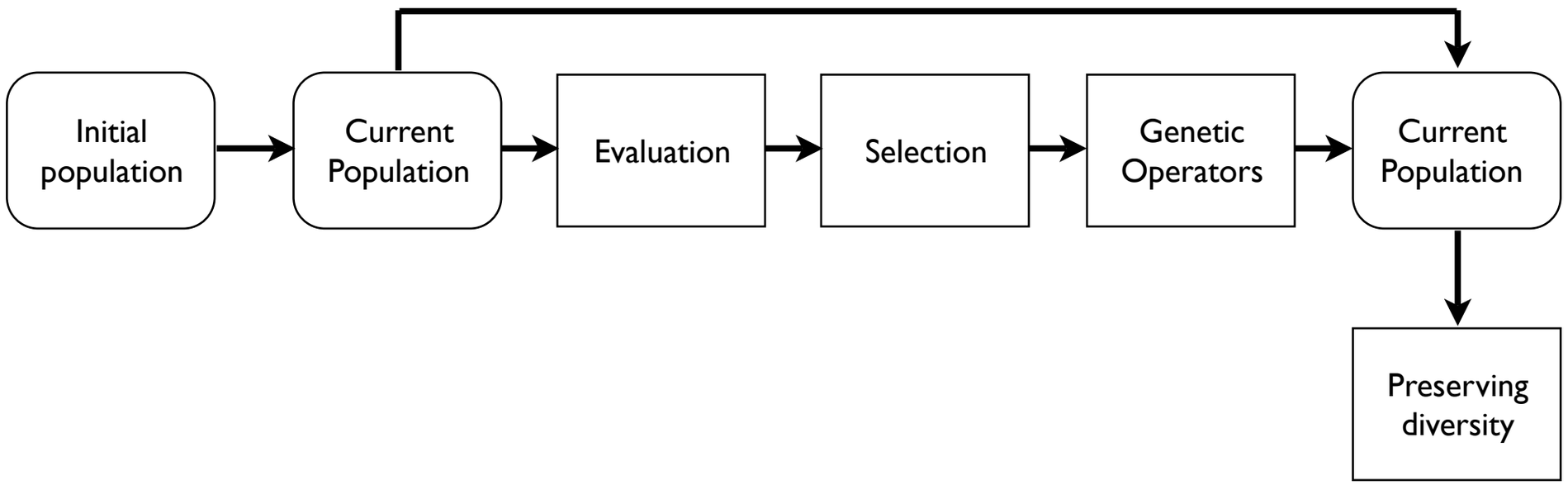


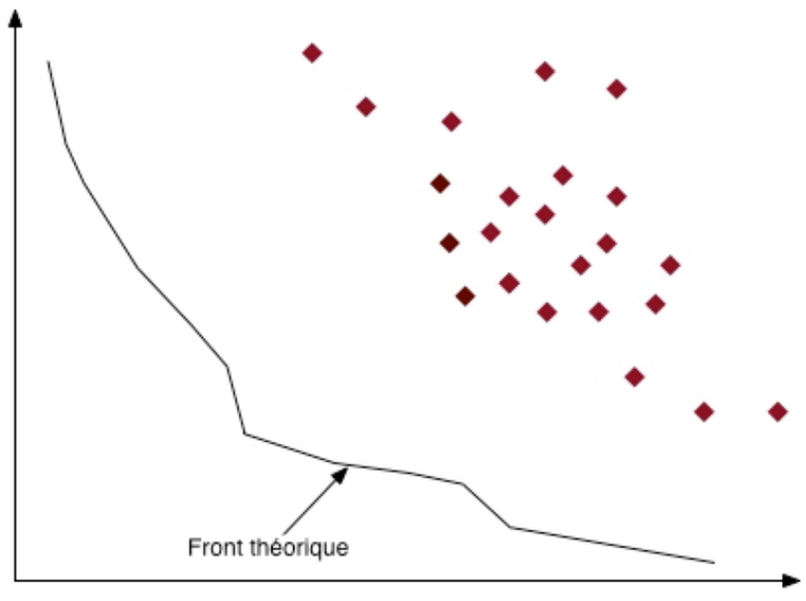
GA optimization: the *orchidée* algorithm



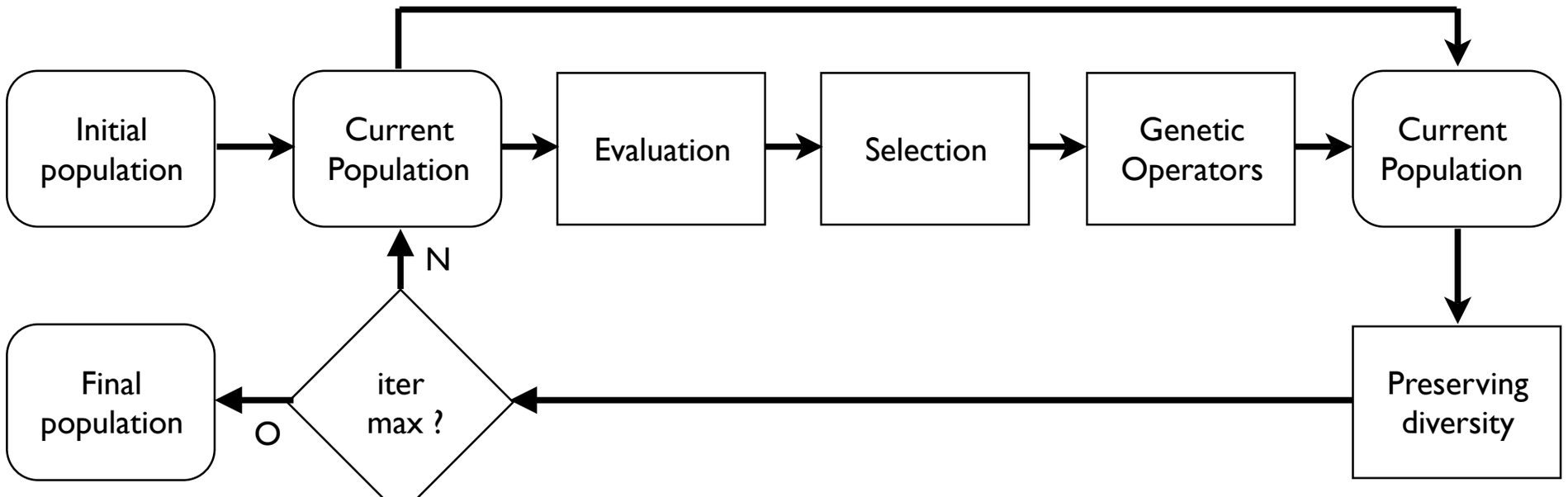


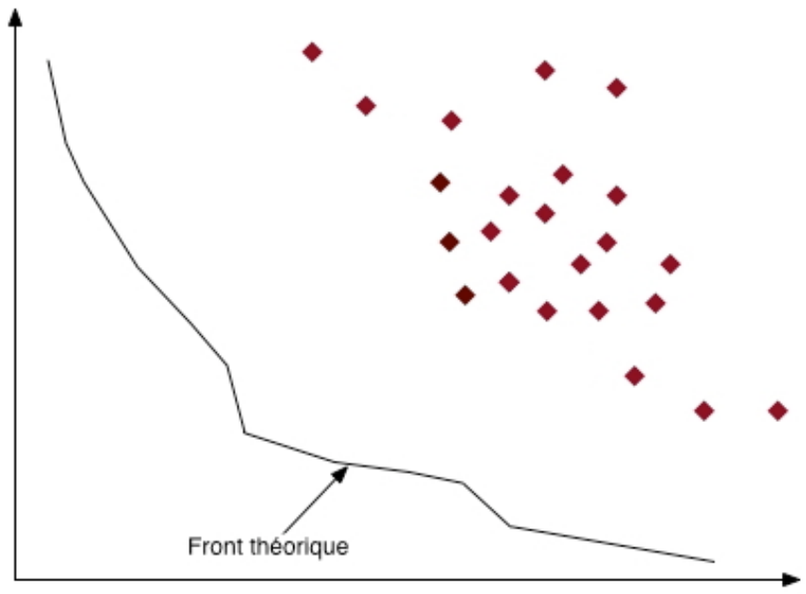
GA optimization: the *orchidée* algorithm



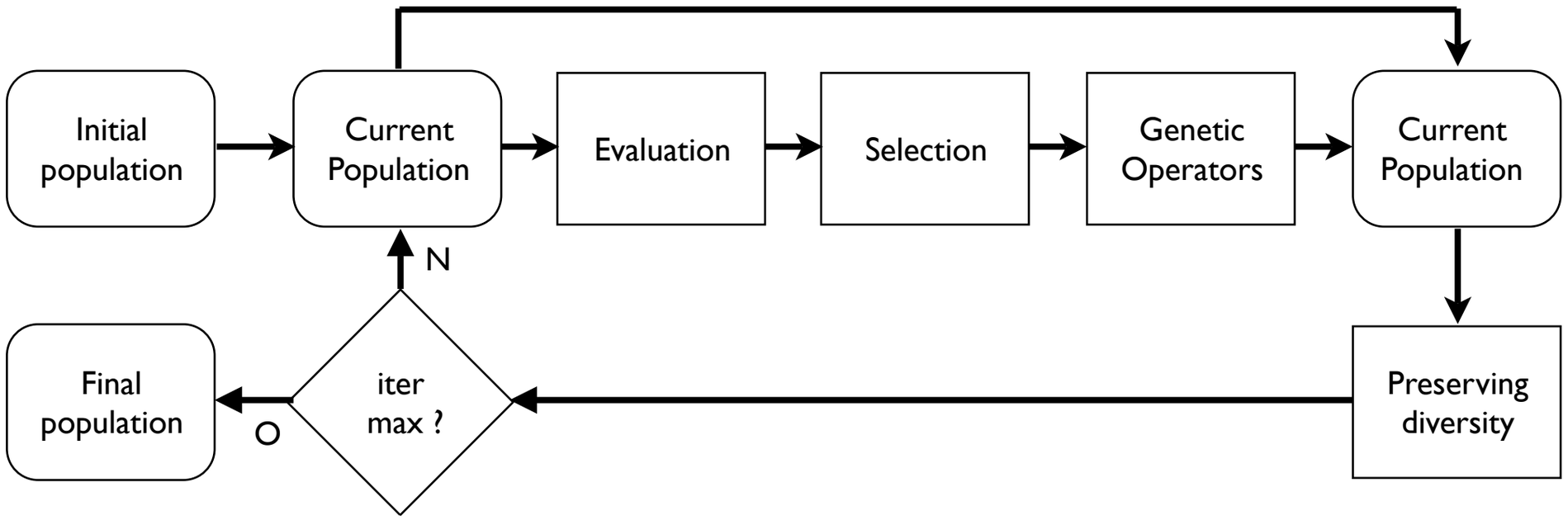


GA optimization: the *orchidée* algorithm

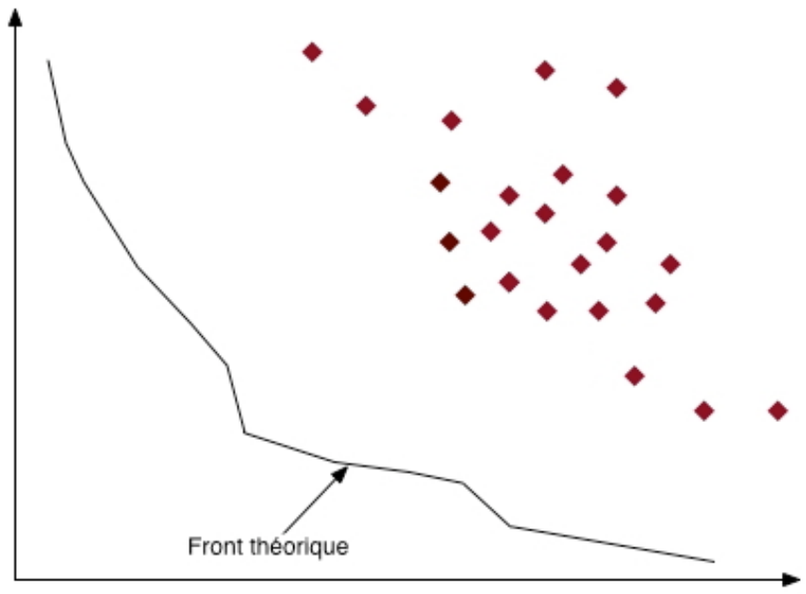




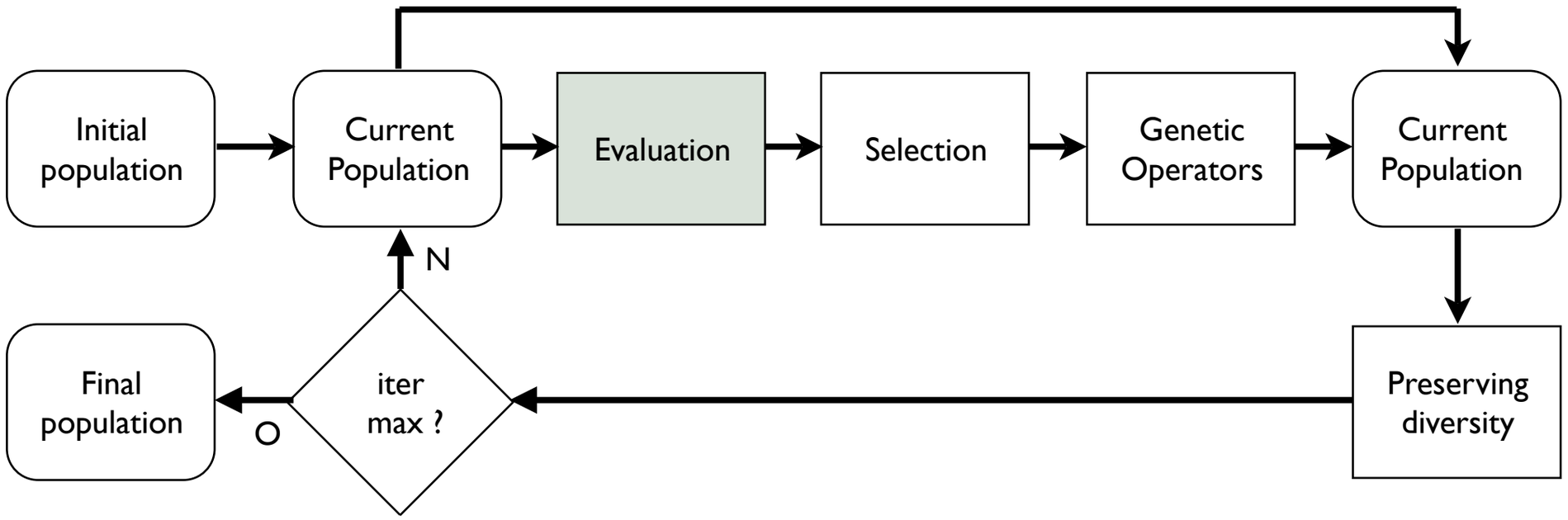
GA optimization: the *orchidée* algorithm



Goals :

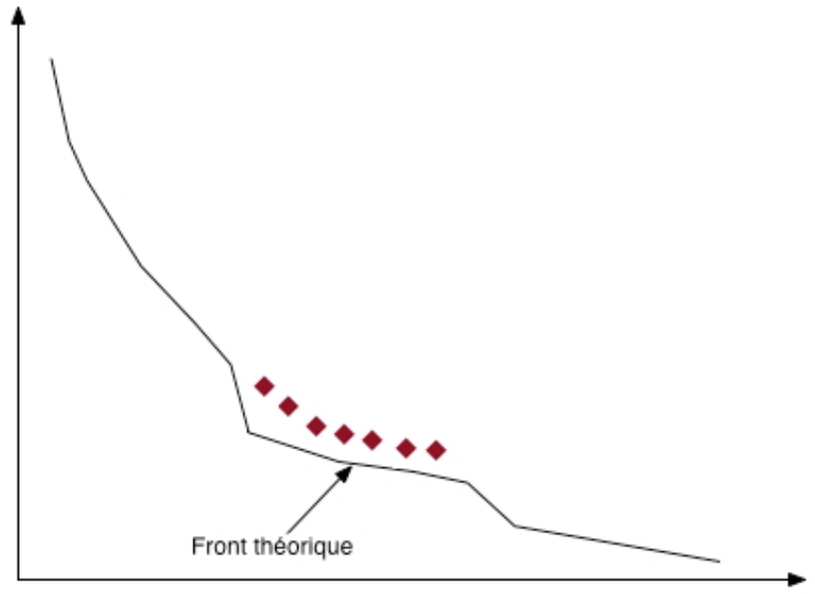


GA optimization: the *orchidée* algorithm

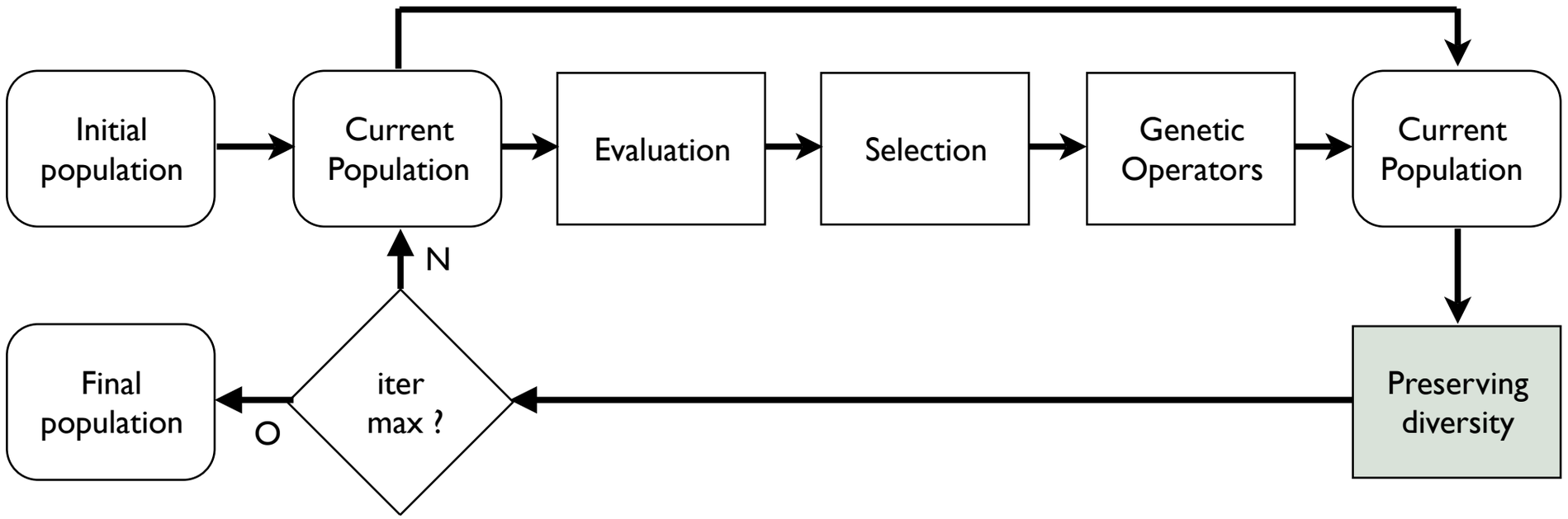


Goals :

- Convergence towards Pareto front

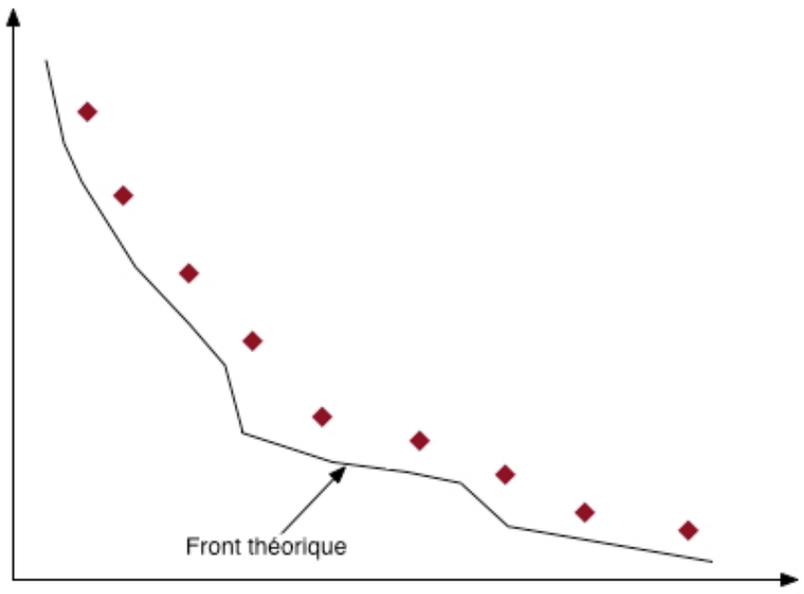


GA optimization: the *orchidée* algorithm



Goals :

- Convergence towards Pareto front
- Diversity along the Pareto front

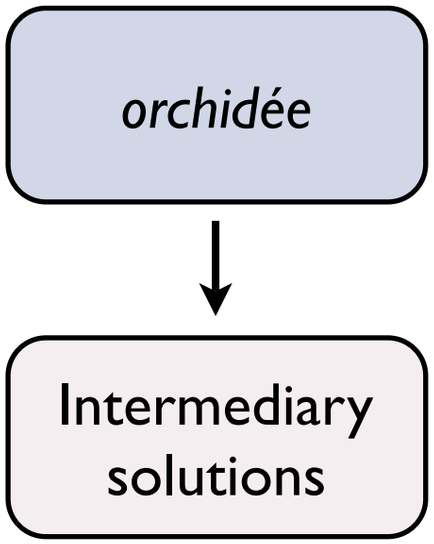


User preferences (1)

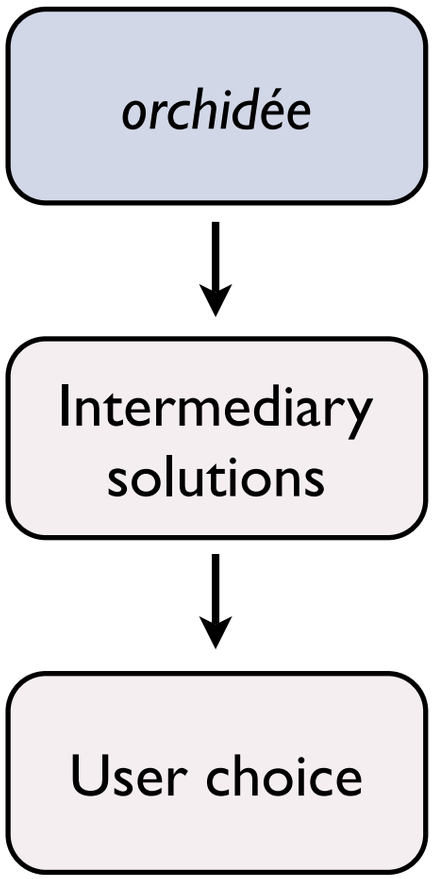
User preferences (1)

orchidée

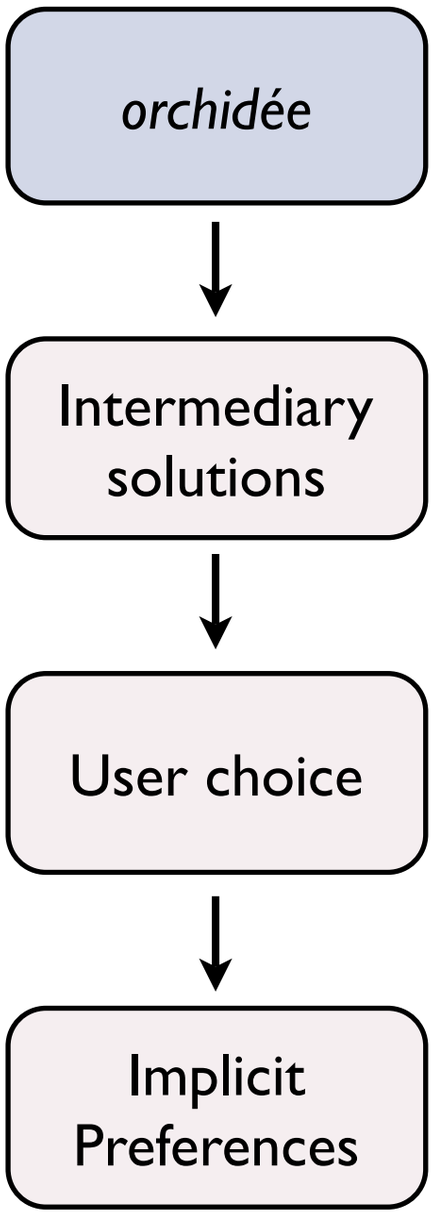
User preferences (1)



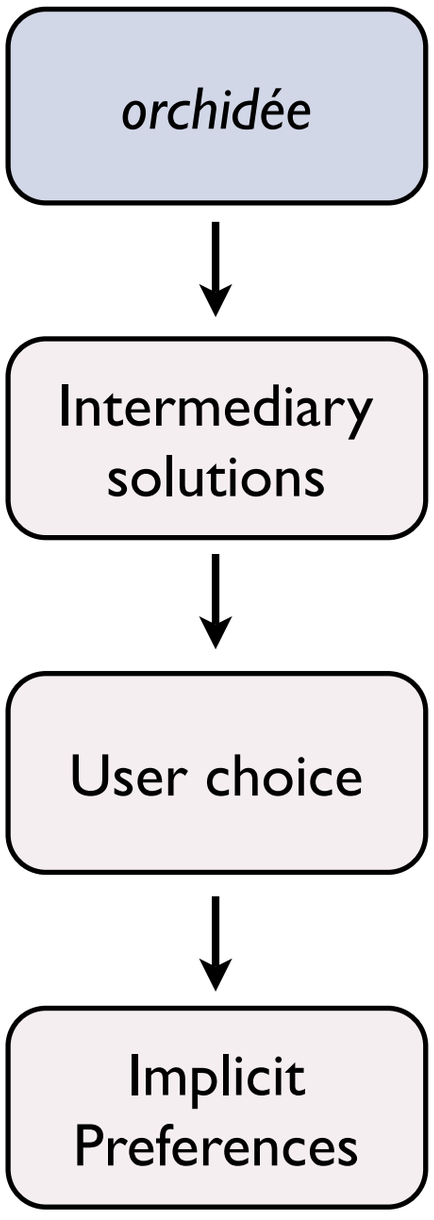
User preferences (1)

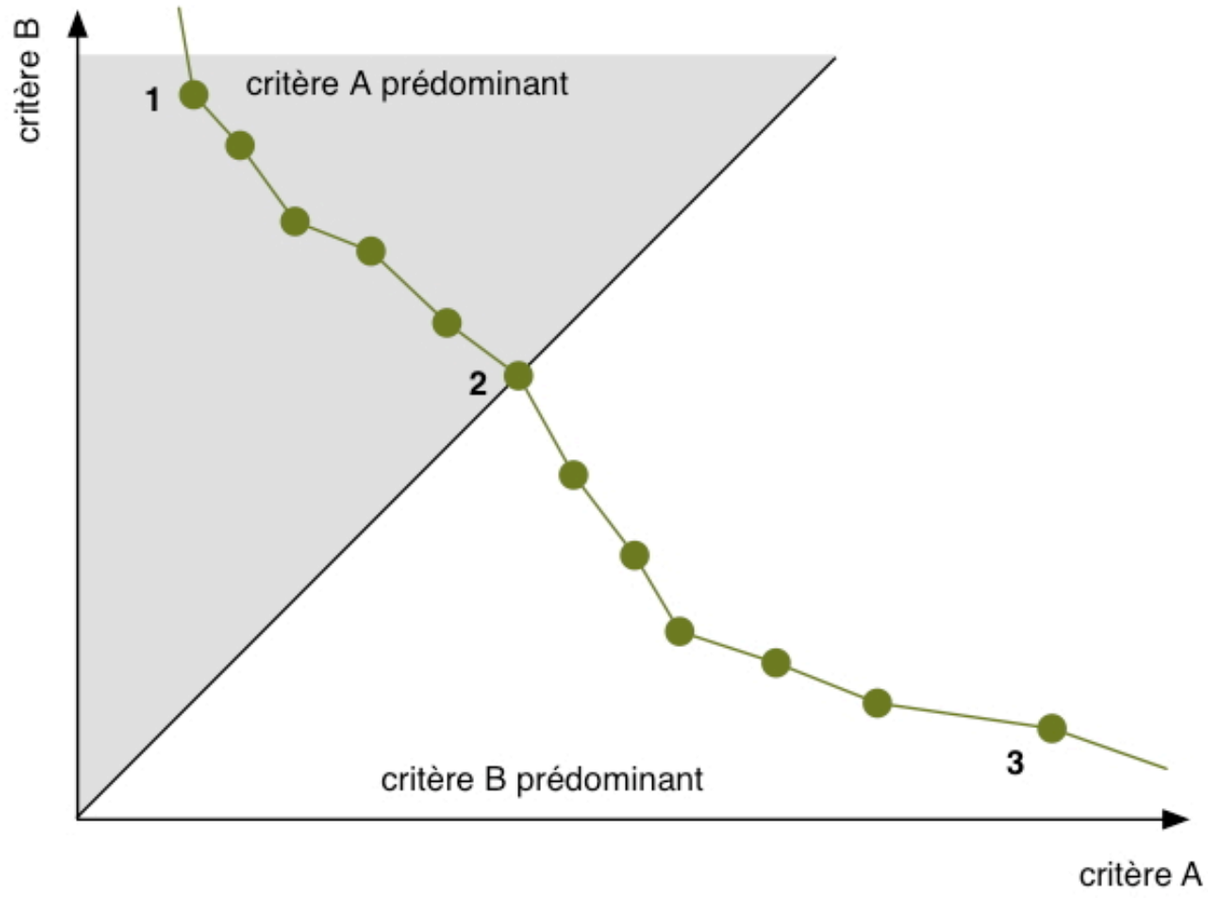


User preferences (1)

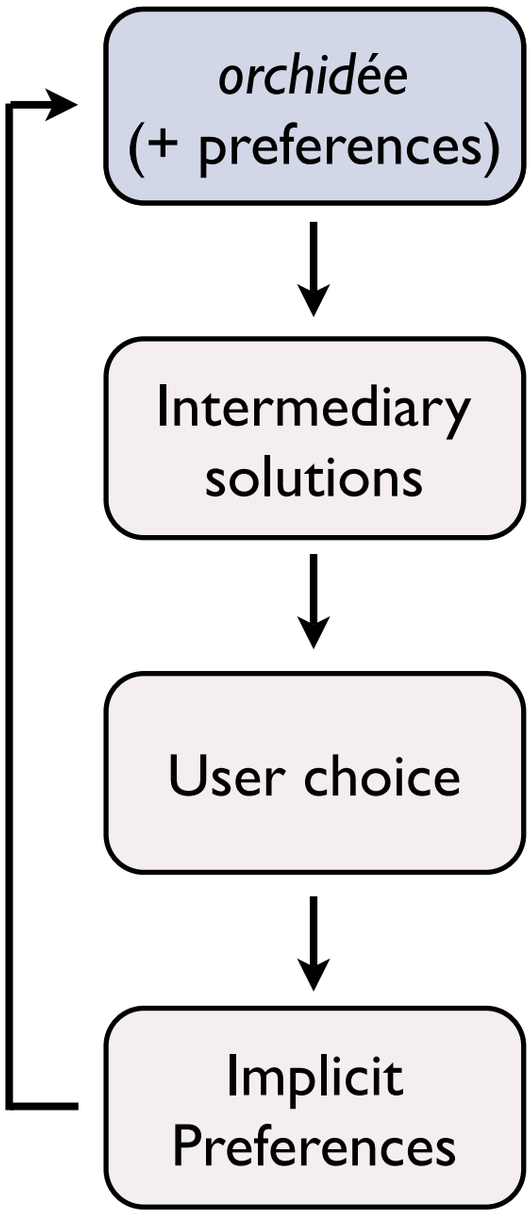


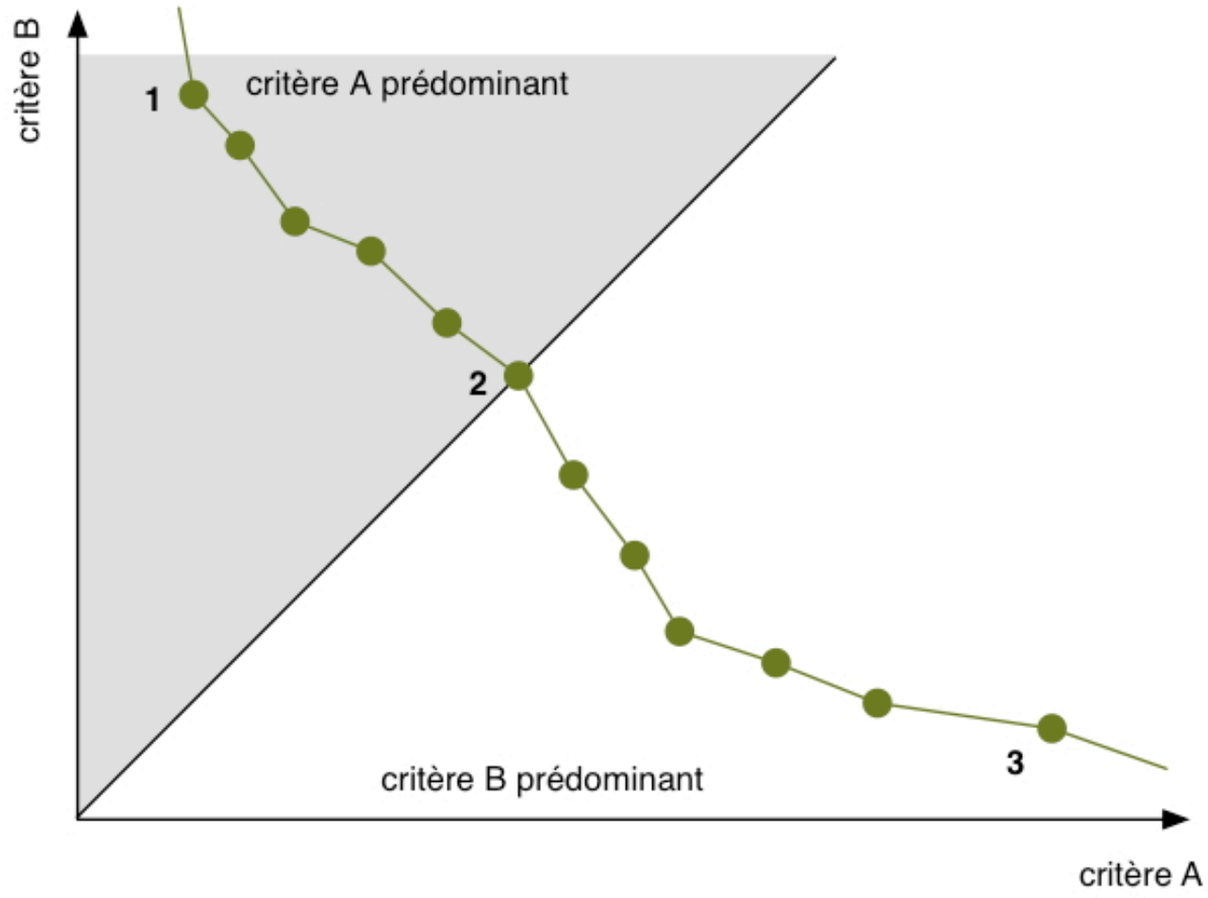
User preferences (1)





User preferences (1)





User preferences (2)

User preferences (2)

- Weighted Chebychev norm [Jaszkiewicz2002] :

$$\|x\|_\lambda = \max_k \lambda_k |x_k|$$

User preferences (2)

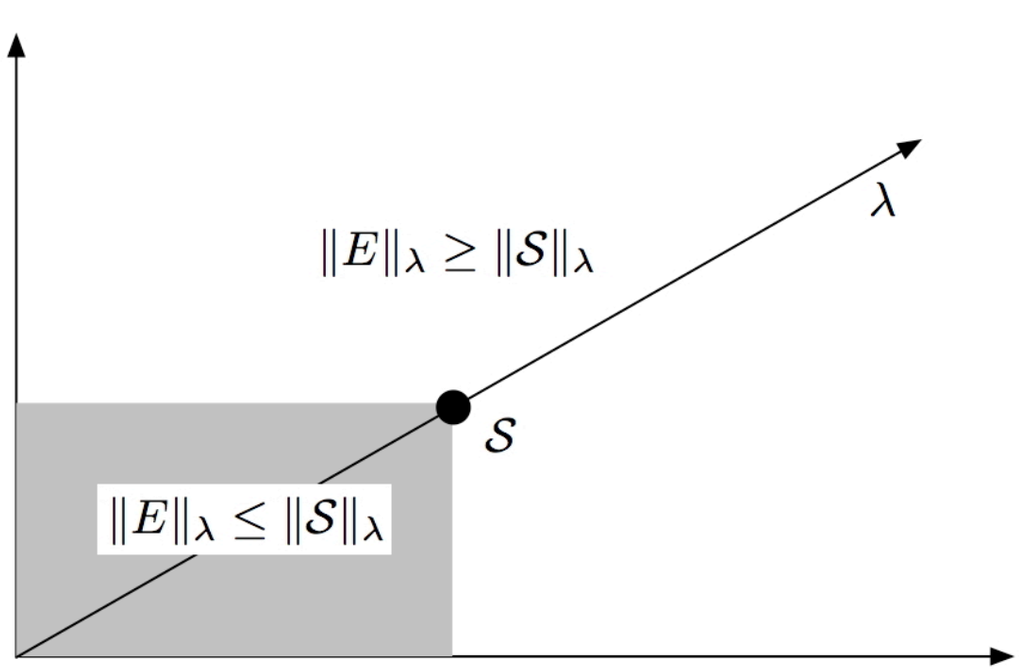
- Weighted Chebychev norm [Jaszkiewicz2002] :

$$\|x\|_\lambda = \max_k \lambda_k |x_k| \qquad \text{soit :} \qquad \|\mathcal{S}\|_\lambda = \max_k \lambda_k D_{\mathcal{T}}^k(\mathcal{S})$$

User preferences (2)

- Weighted Chebychev norm [Jaszkiewicz2002] :

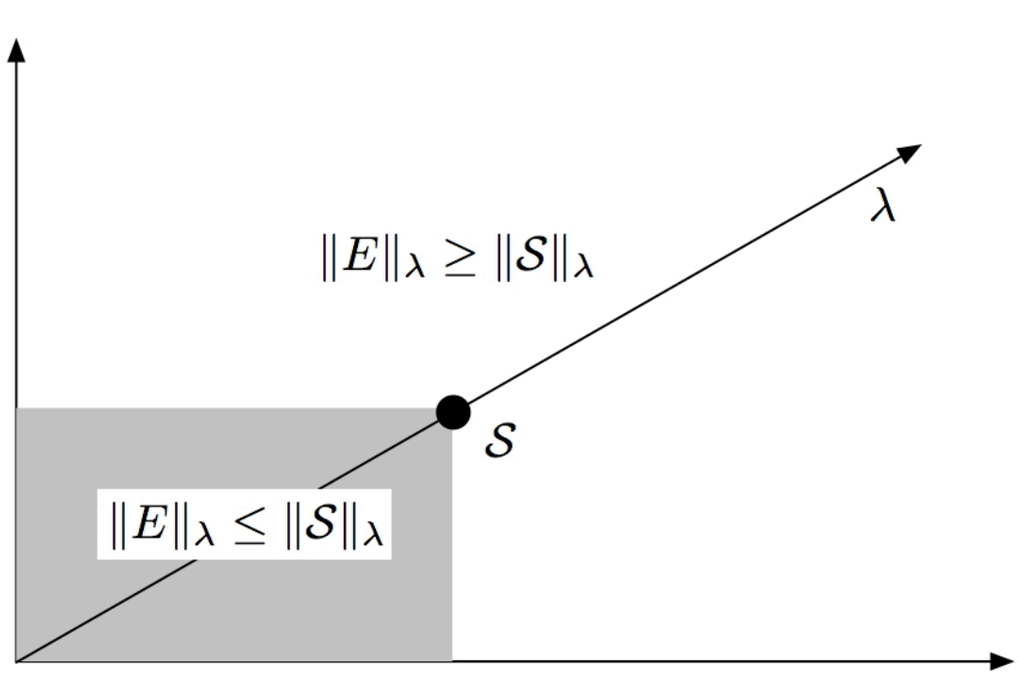
$$\|x\|_\lambda = \max_k \lambda_k |x_k| \qquad \text{soit :} \qquad \|\mathcal{S}\|_\lambda = \max_k \lambda_k D_T^k(\mathcal{S})$$

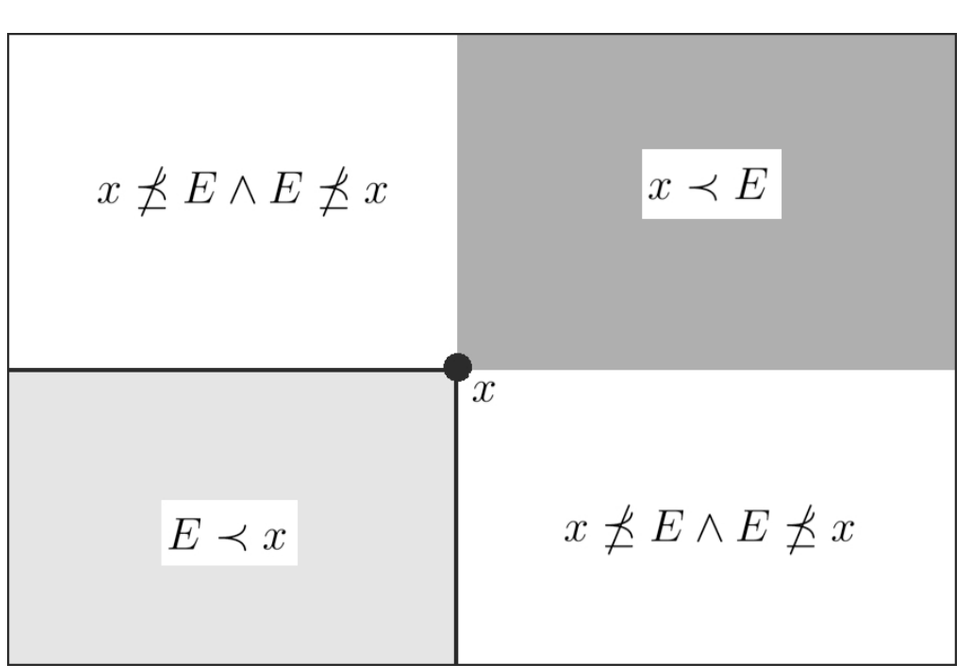


User preferences (2)

- Weighted Chebychev norm [Jaszkiewicz2002] :

$$\|x\|_\lambda = \max_k \lambda_k |x_k| \qquad \text{soit :} \qquad \|\mathcal{S}\|_\lambda = \max_k \lambda_k D_T^k(\mathcal{S})$$

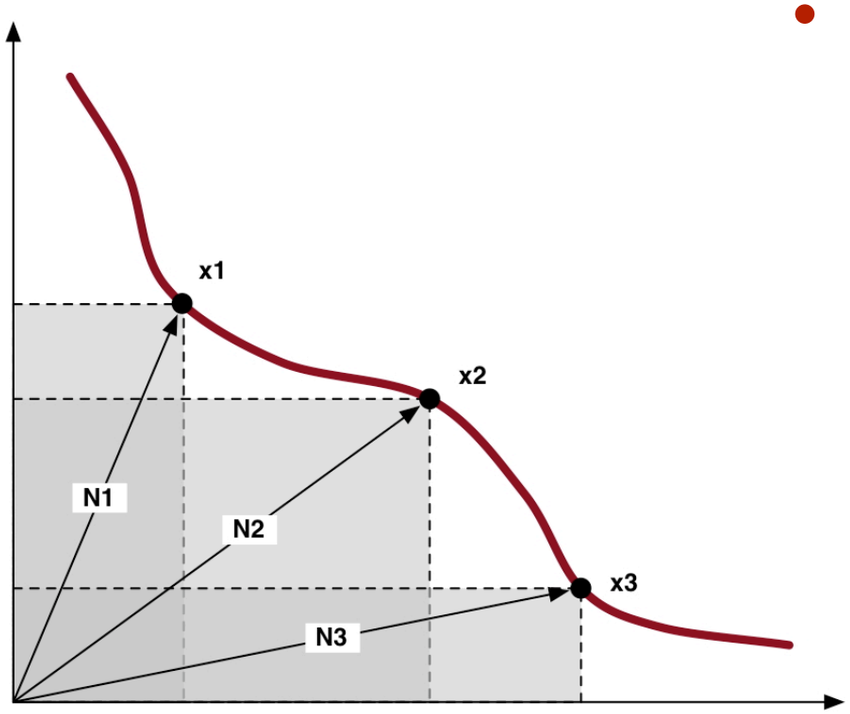




User preferences (3)

User preferences (3)

- Each efficient solution corresponds to a weight set

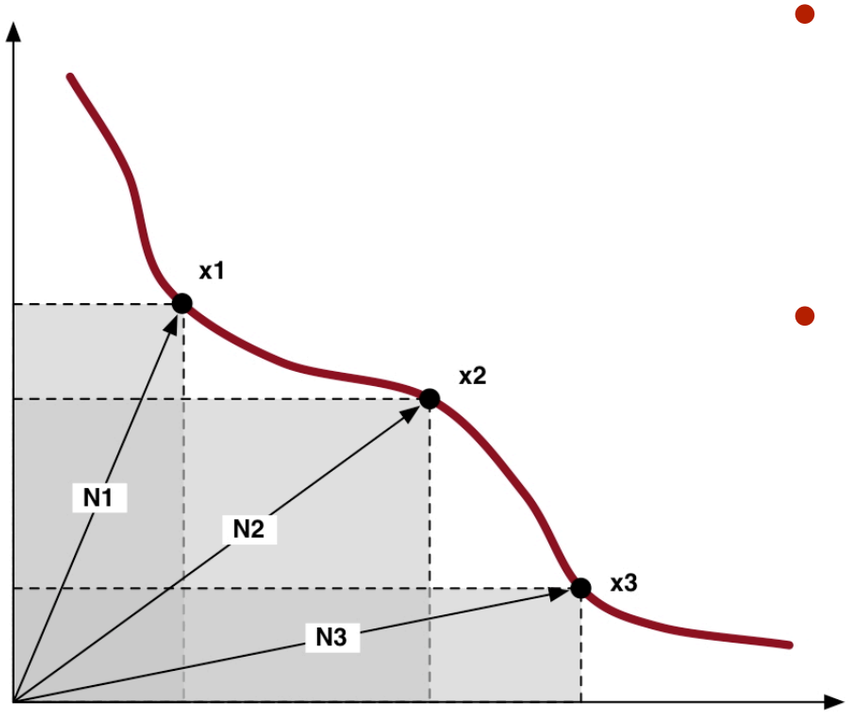


- Fundamental property [Steuer1986] :

$$\mathcal{S}^* \in \mathcal{P}^* \Leftrightarrow \exists \lambda,\ \mathcal{S}^* = \underset{\mathcal{S}}{\operatorname{argmin}}\ \|\mathcal{S}\|_\lambda$$

User preferences (3)

- Each efficient solution corresponds to a weight set



- Fundamental property [Steuer1986] :

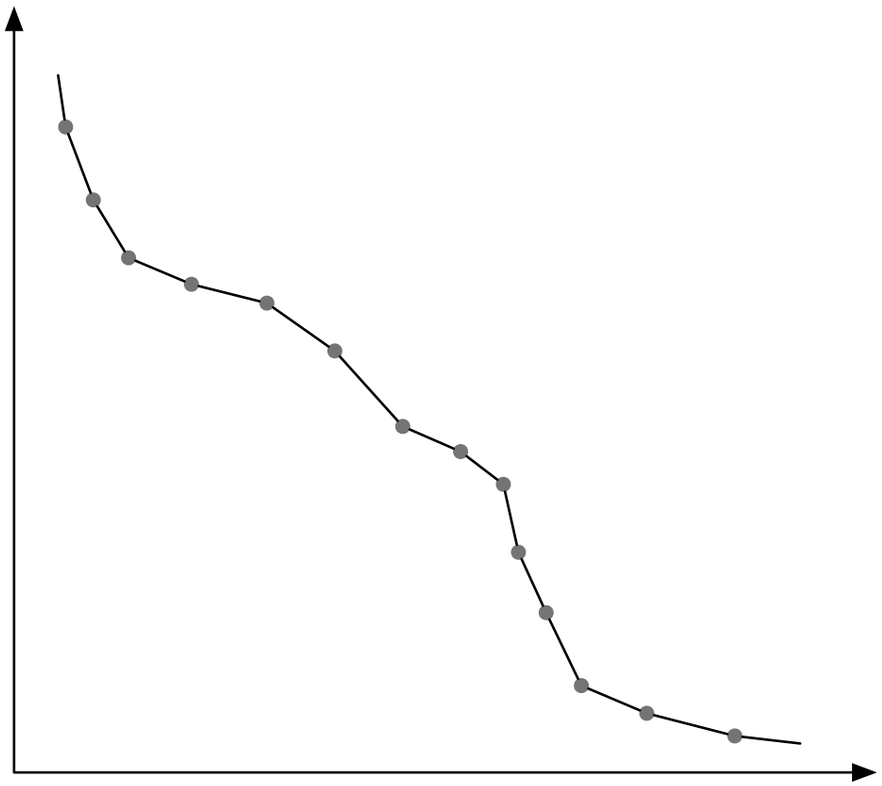
$$\mathcal{S}^* \in \mathcal{P}^* \Leftrightarrow \exists \lambda,\ \mathcal{S}^* = \underset{\mathcal{S}}{\operatorname{argmin}}\ \|\mathcal{S}\|_\lambda$$

- Computing weights from criteria [Carpentier2008] :

$$\lambda_k = \frac{\prod_{j \neq k}^{K} x_j}{\sum_{i=1}^{K} \prod_{j \neq i} x_j}\ ,\ k = 1, ..., K$$

User preferences (3)

- Each efficient solution corresponds to a weight set



Fundamental property [Steuer1986] :

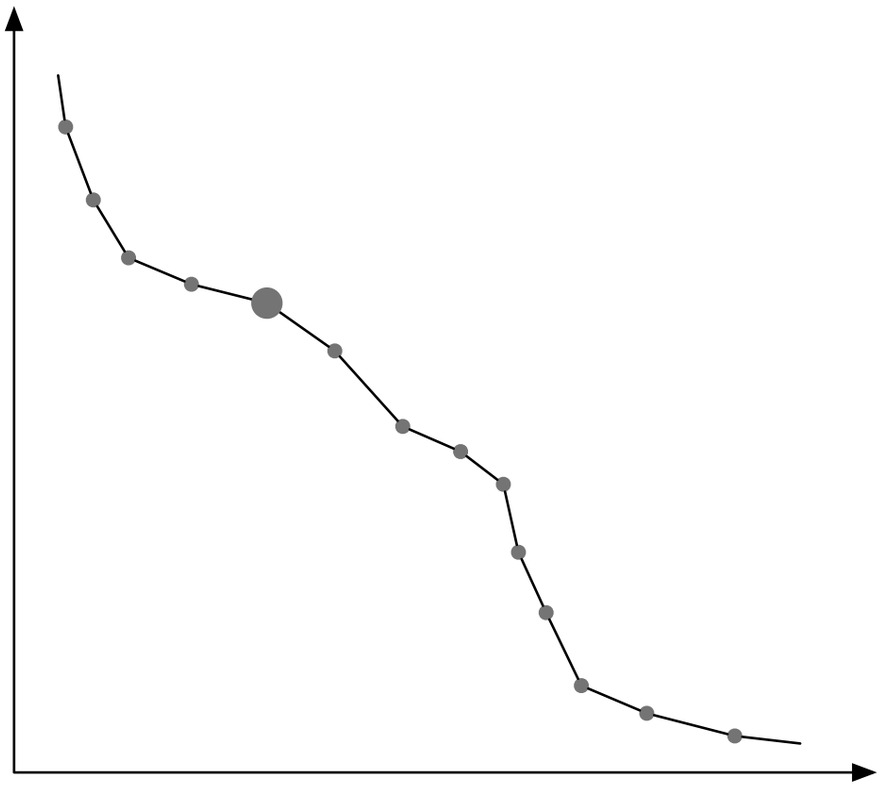
$$\mathcal{S}^* \in \mathcal{P}^* \Leftrightarrow \exists \lambda,\ \mathcal{S}^* = \underset{\mathcal{S}}{\operatorname{argmin}}\ \|\mathcal{S}\|_\lambda$$

Computing weights from criteria [Carpentier2008] :

$$\lambda_k = \frac{\prod_{j \neq k} x_j}{\sum_{i=1}^{K} \prod_{j \neq i} x_j},\ k = 1, \ldots, K$$

User preferences (3)

- Each efficient solution corresponds to a weight set



Fundamental property [Steuer1986] :

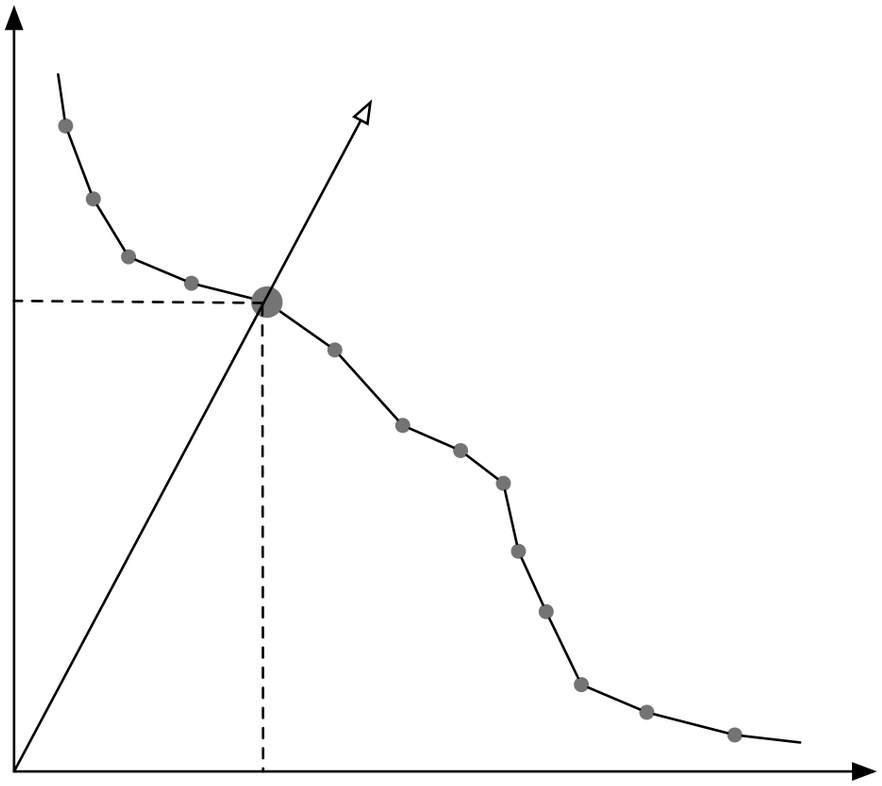
$$\mathcal{S}^* \in \mathcal{P}^* \Leftrightarrow \exists \lambda,\ \mathcal{S}^* = \underset{\mathcal{S}}{\operatorname{argmin}}\ \|\mathcal{S}\|_\lambda$$

Computing weights from criteria [Carpentier2008] :

$$\lambda_k = \frac{\prod_{j \neq k} x_j}{\sum_{i=1}^{K} \prod_{j \neq i} x_j},\ k = 1, \ldots, K$$

User preferences (3)

- Each efficient solution corresponds to a weight set



Fundamental property [Steuer1986] :

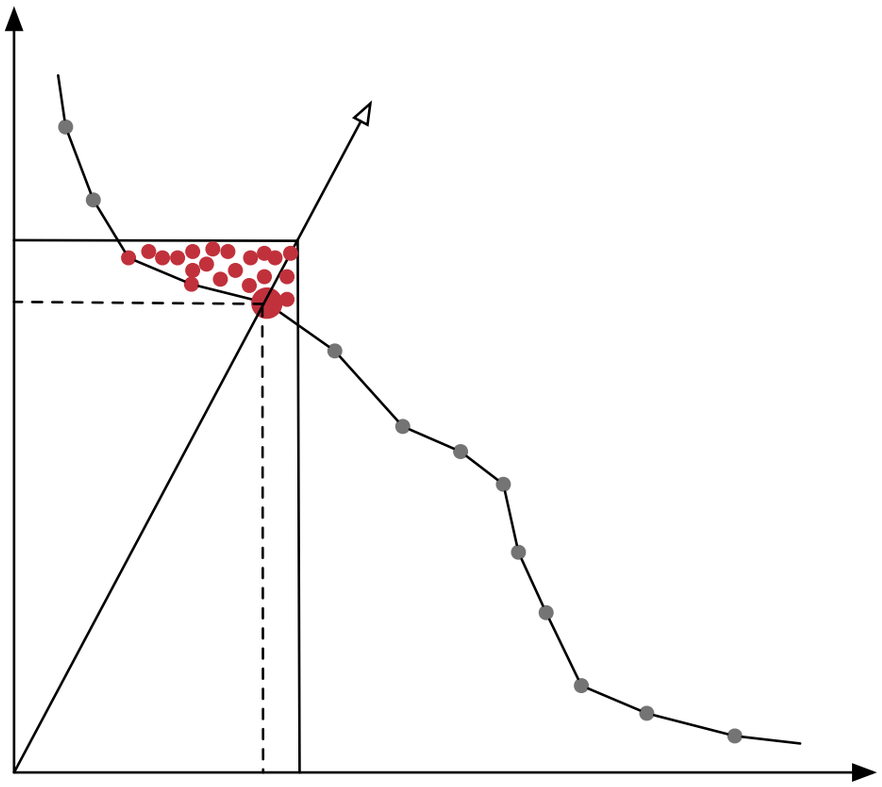
$$\mathcal{S}^* \in \mathcal{P}^* \Leftrightarrow \exists \lambda,\ \mathcal{S}^* = \underset{\mathcal{S}}{\operatorname{argmin}}\ \|\mathcal{S}\|_\lambda$$

Computing weights from criteria [Carpentier2008] :

$$\lambda_k = \frac{\prod_{j \neq k}^{K} x_j}{\sum_{i=1}^{K} \prod_{j \neq i} x_j},\ k = 1, ..., K$$

User preferences (3)

- Each efficient solution corresponds to a weight set



Fundamental property [Steuer1986] :

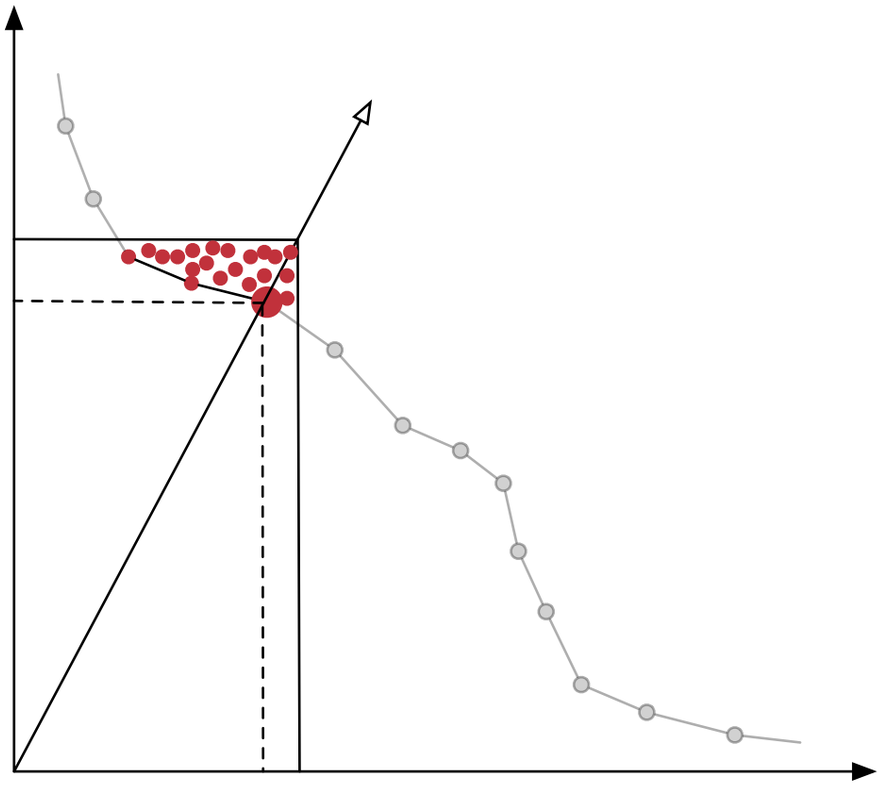
$$\mathcal{S}^* \in \mathcal{P}^* \Leftrightarrow \exists \lambda,\ \mathcal{S}^* = \underset{\mathcal{S}}{\operatorname{argmin}}\ \|\mathcal{S}\|_\lambda$$

Computing weights from criteria [Carpentier2008] :

$$\lambda_k = \frac{\prod_{j \neq k}^{K} x_j}{\sum_{i=1}^{K} \prod_{j \neq i} x_j},\ k = 1, ..., K$$

User preferences (3)

- Each efficient solution corresponds to a weight set



Fundamental property [Steuer1986] :

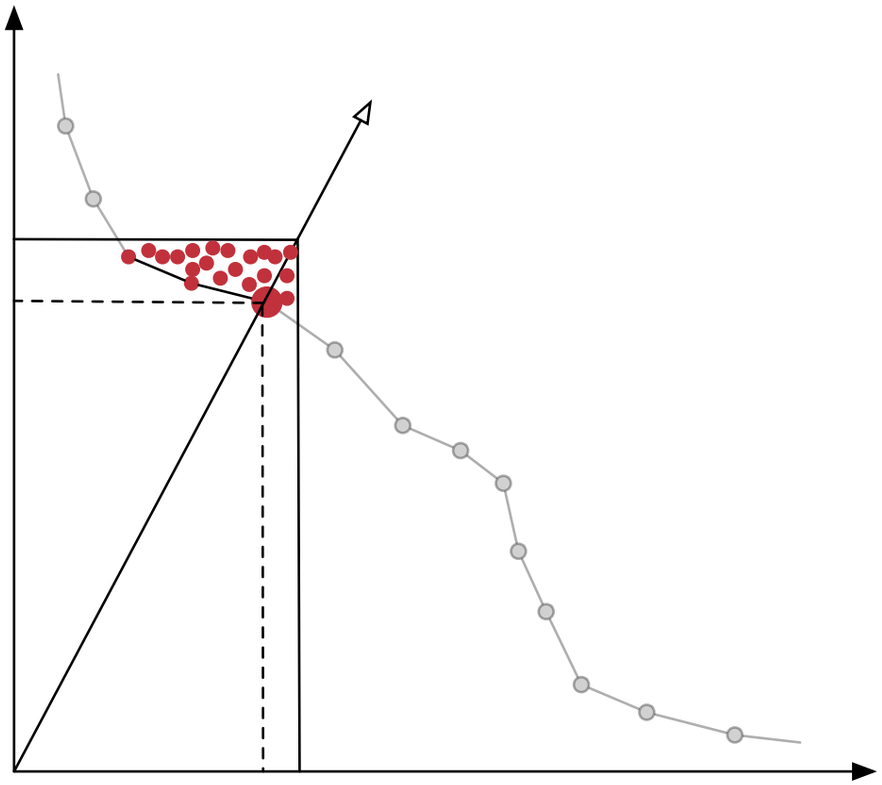
$$\mathcal{S}^* \in \mathcal{P}^* \Leftrightarrow \exists \lambda,\ \mathcal{S}^* = \underset{\mathcal{S}}{\operatorname{argmin}}\ \|\mathcal{S}\|_\lambda$$

Computing weights from criteria [Carpentier2008] :

$$\lambda_k = \frac{\prod_{j \neq k} x_j}{\sum_{i=1}^{K} \prod_{j \neq i} x_j},\ k = 1, ..., K$$

User preferences (3)

- Each efficient solution corresponds to a weight set



Fundamental property [Steuer1986] :

$$\mathcal{S}^* \in \mathcal{P}^* \Leftrightarrow \exists \lambda,\ \mathcal{S}^* = \underset{\mathcal{S}}{\operatorname{argmin}}\ \|\mathcal{S}\|_\lambda$$

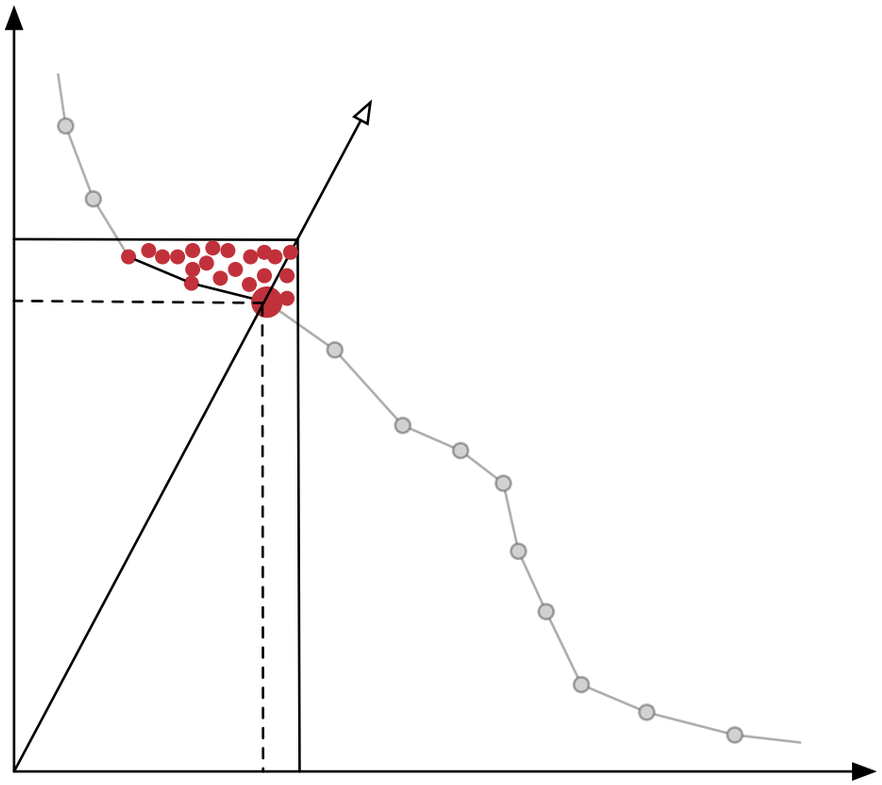
Computing weights from criteria [Carpentier2008] :

$$\lambda_k = \frac{\prod_{j \neq k} x_j}{\sum_{i=1}^{K} \prod_{j \neq i} x_j},\ k = 1, ..., K$$

- *Orchidée* computes configurations fitness thanks to a Chebychev norm

User preferences (3)

- Each efficient solution corresponds to a weight set



Fundamental property [Steuer1986] :

$$\mathcal{S}^* \in \mathcal{P}^* \Leftrightarrow \exists \lambda,\ \mathcal{S}^* = \underset{\mathcal{S}}{\operatorname{argmin}}\ \|\mathcal{S}\|_\lambda$$

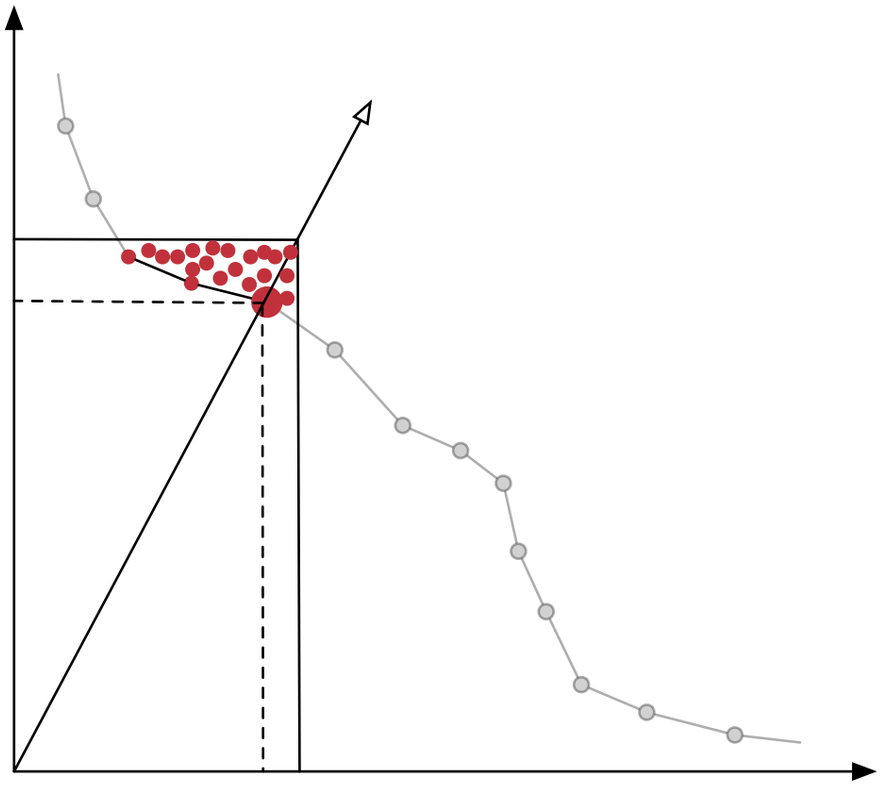
Computing weights from criteria [Carpentier2008] :

$$\lambda_k = \frac{\prod_{j \neq k} x_j}{\sum_{i=1}^{K} \prod_{j \neq i} x_j},\ k = 1, ..., K$$

- *Orchidée* computes configurations fitness thanks to a Chebychev norm
- When preferences are unknown weights are randomly drawn at each generation (multiobjective optimization)

User preferences (3)

- Each efficient solution corresponds to a weight set



Fundamental property [Steuer1986] :

$$\mathcal{S}^* \in \mathcal{P}^* \Leftrightarrow \exists \lambda,\ \mathcal{S}^* = \underset{\mathcal{S}}{\operatorname{argmin}}\ \|\mathcal{S}\|_\lambda$$

Computing weights from criteria [Carpentier2008] :

$$\lambda_k = \frac{\prod_{j \neq k} x_j}{\sum_{i=1}^{K} \prod_{j \neq i} x_j},\ k = 1, ..., K$$

- *Orchidée* computes configurations fitness thanks to a Chebychev norm
- When preferences are unknown weights are randomly drawn at each generation (multiobjective optimization)
- When preferences are known weights are fixed (monoobjective optimization)

Contents

Compositional context

Compositional context

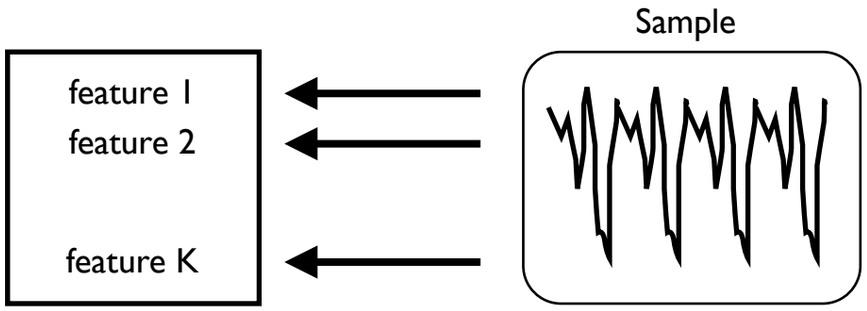
- Any musical material comes within a musical gesture

Compositional context

- Any musical material comes within a musical gesture
- Depending on this gesture all configurations may not be feasible

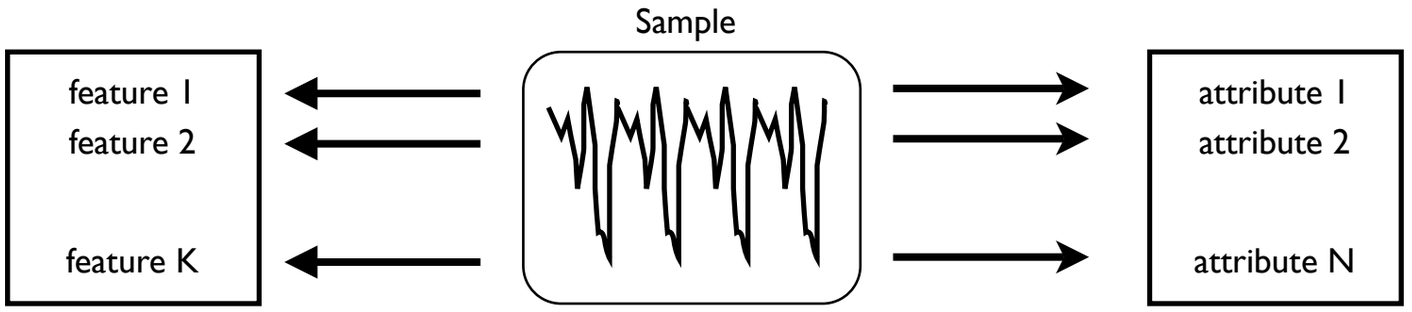
Compositional context

- Any musical material comes within a musical gesture
- Depending on this gesture all configurations may not be feasible



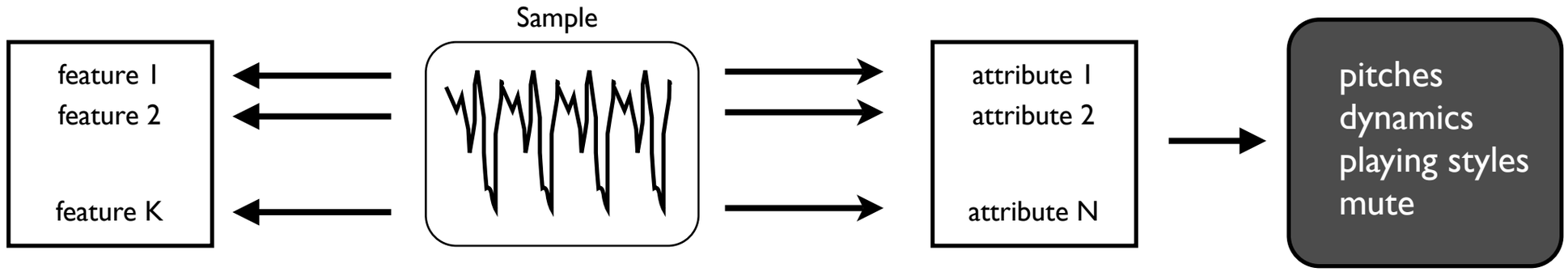
Compositional context

- Any musical material comes within a musical gesture
- Depending on this gesture all configurations may not be feasible



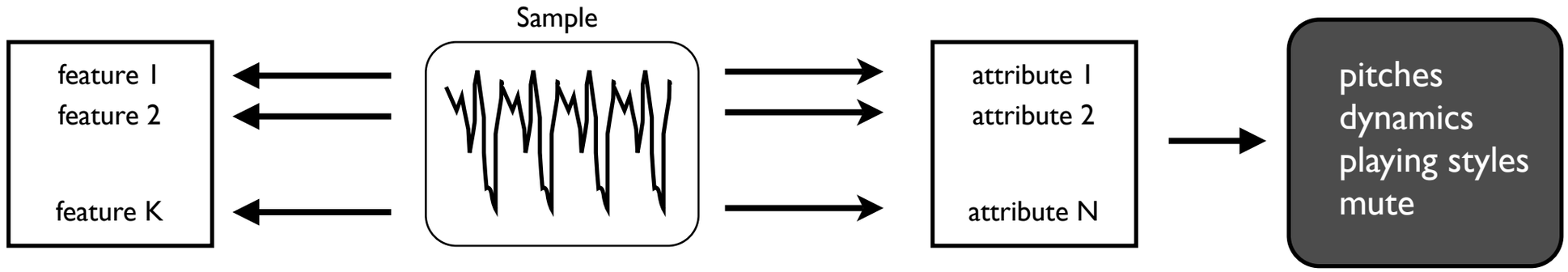
Compositional context

- Any musical material comes within a musical gesture
- Depending on this gesture all configurations may not be feasible



Compositional context

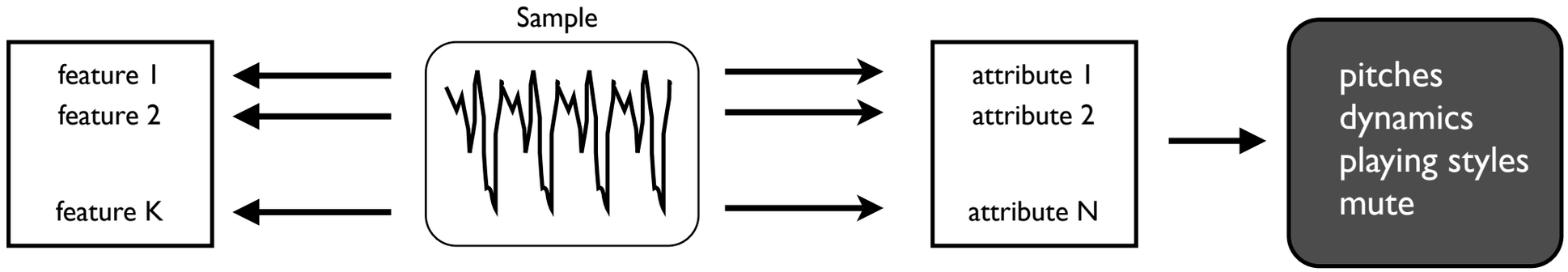
- Any musical material comes within a musical gesture
- Depending on this gesture all configurations may not be feasible



- Modeling context with ***global constraints*** on attributes :

Compositional context

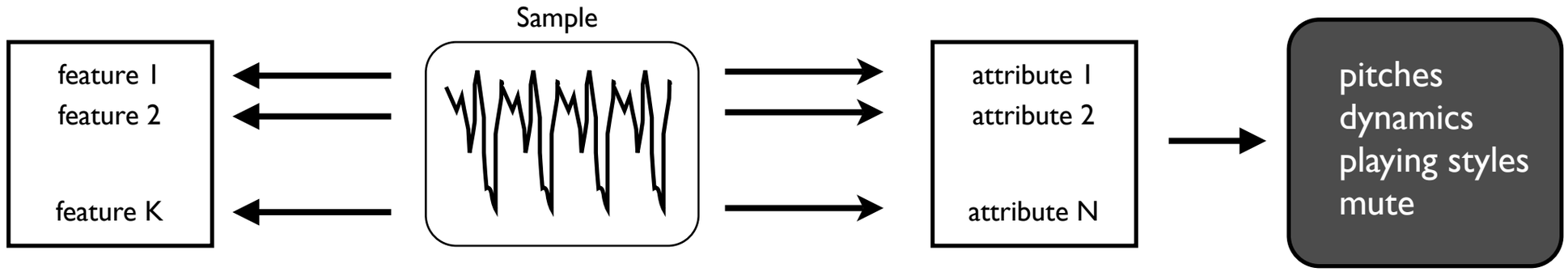
- Any musical material comes within a musical gesture
- Depending on this gesture all configurations may not be feasible



- Modeling context with ***global constraints*** on attributes :
  - “Between 8 et 12 instruments”

Compositional context

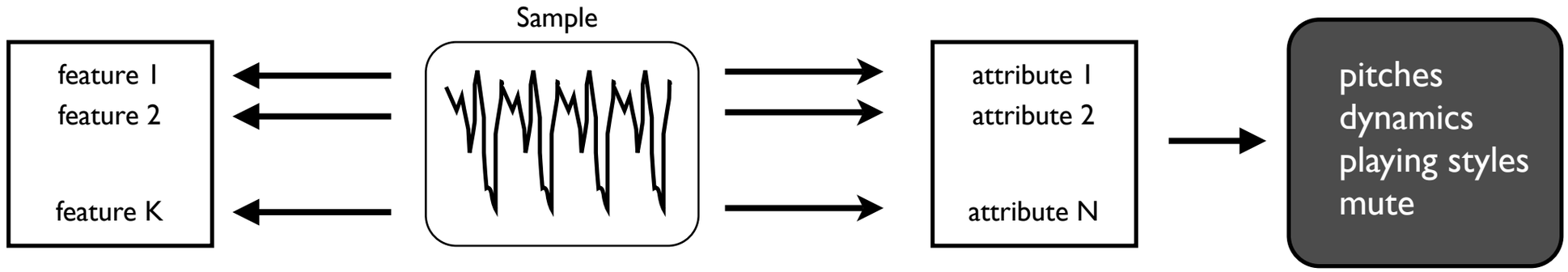
- Any musical material comes within a musical gesture
- Depending on this gesture all configurations may not be feasible



- Modeling context with ***global constraints*** on attributes :
  - “Between 8 et 12 instruments”
  - “No more than 3 fortissimo”

Compositional context

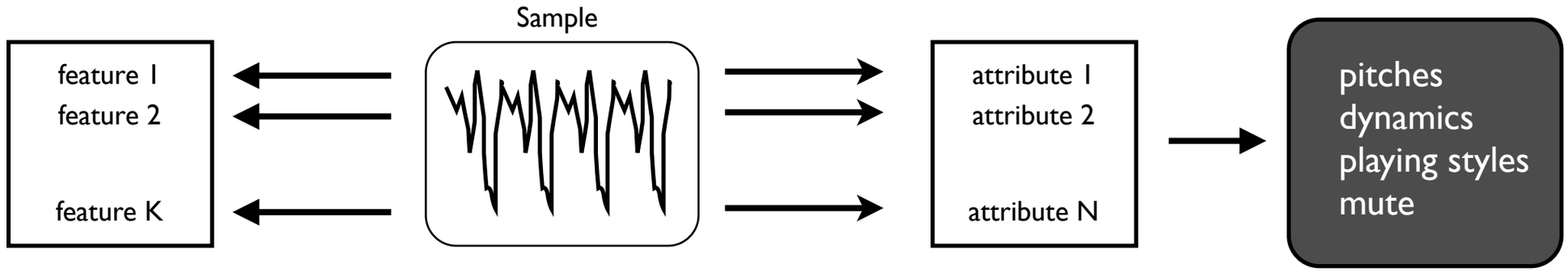
- Any musical material comes within a musical gesture
- Depending on this gesture all configurations may not be feasible



- Modeling context with ***global constraints*** on attributes :
  - “Between 8 et 12 instruments”
  - “No more than 3 fortissimo”
  - “At least two different pitches”

Compositional context

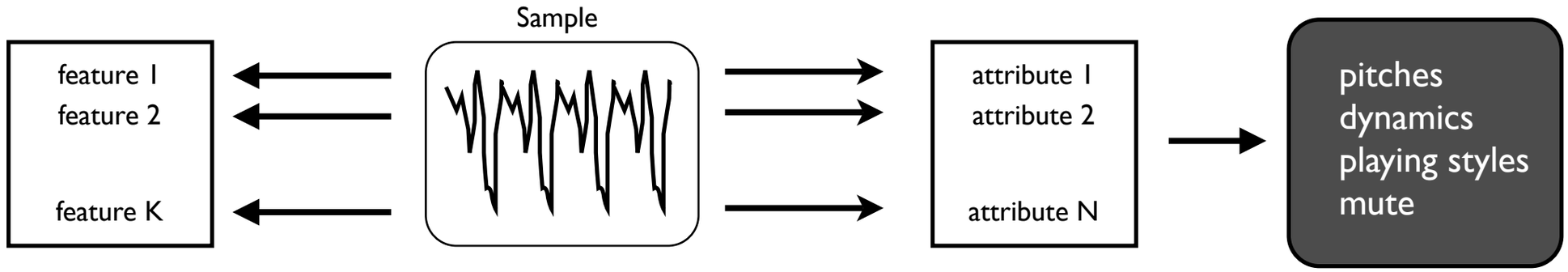
- Any musical material comes within a musical gesture
- Depending on this gesture all configurations may not be feasible



- Modeling context with ***global constraints*** on attributes :
  - “Between 8 et 12 instruments”
  - “No more than 3 fortissimo”
  - “At least two different pitches”
  - “All strings play with a mute”

Compositional context

- Any musical material comes within a musical gesture
- Depending on this gesture all configurations may not be feasible



- Modeling context with ***global constraints*** on attributes :
  - “Between 8 et 12 instruments”
  - “No more than 3 fortissimo”
  - “At least two different pitches”
  - “All strings play with a mute”
- ***Local search / soft constraints*** approach: Iteratively update a single configuration to minimize a set of cost functions

Design / Conflict?

Design / Conflict?

- Design constraints: Anything ***required*** in the orchestration

Design / Conflict?

- Design constraints: Anything ***required*** in the orchestration
- Conflict constraints: Anything to ***avoid*** in the orchestration

Design / Conflict?

- Design constraints: Anything ***required*** in the orchestration
- Conflict constraints: Anything to ***avoid*** in the orchestration

- Design constraints may be satisfied by ***instantiating free*** variables

Design / Conflict?

- Design constraints: Anything ***required*** in the orchestration
- Conflict constraints: Anything to ***avoid*** in the orchestration

- Design constraints may be satisfied by ***instantiating free*** variables
- Conflict constraints may be satisfied by ***freeing instantiated*** variables

CDCSolver

CDCSolver

3 neighborhood heuristics:

CDCSolver

3 neighborhood heuristics:

1. If a *conflict* constraint is violated, first update an instantiated variable.

CDCSolver

3 neighborhood heuristics:

1. If a *conflict* constraint is violated, first update an instantiated variable.
2. If a *design* constraint is violated, first update a free variable.

CDCSolver

3 neighborhood heuristics:

1. If a *conflict* constraint is violated, first update an instantiated variable.
2. If a *design* constraint is violated, first update a free variable.
3. There is a priority variable to change [Codognet2002]. Decision rule is based on a *min-conflict* heuristic.

CDCSolver

3 neighborhood heuristics:

1. If a *conflict* constraint is violated, first update an instantiated variable.
2. If a *design* constraint is violated, first update a free variable.
3. There is a priority variable to change [Codognet2002]. Decision rule is based on a *min-conflict* heuristic.

1 move heuristic:

CDCSolver

3 neighborhood heuristics:

1. If a *conflict* constraint is violated, first update an instantiated variable.
2. If a *design* constraint is violated, first update a free variable.
3. There is a priority variable to change [Codognet2002]. Decision rule is based on a *min-conflict* heuristic.

1 move heuristic:

4. In the current neighborhood choose the configuration that minimizes the global cost function (*min-conflict*).

CDCSolver

3 neighborhood heuristics:

1. If a *conflict* constraint is violated, first update an instantiated variable.
2. If a *design* constraint is violated, first update a free variable.
3. There is a priority variable to change [Codognet2002]. Decision rule is based on a *min-conflict* heuristic.

1 move heuristic:

4. In the current neighborhood choose the configuration that minimizes the global cost function (*min-conflict*).

1 cycle handling heuristic:

CDCSolver

3 neighborhood heuristics:

1. If a *conflict* constraint is violated, first update an instantiated variable.
2. If a *design* constraint is violated, first update a free variable.
3. There is a priority variable to change [Codognet2002]. Decision rule is based on a *min-conflict* heuristic.

1 move heuristic:

4. In the current neighborhood choose the configuration that minimizes the global cost function (*min-conflict*).

1 cycle handling heuristic:

5. Handle cycles and local minima with a short term memory scheme (tabu list [Glover1997])

Combining *orchidée* and *cdcsolver*

Combining *orchidée* and *cdcsolver*

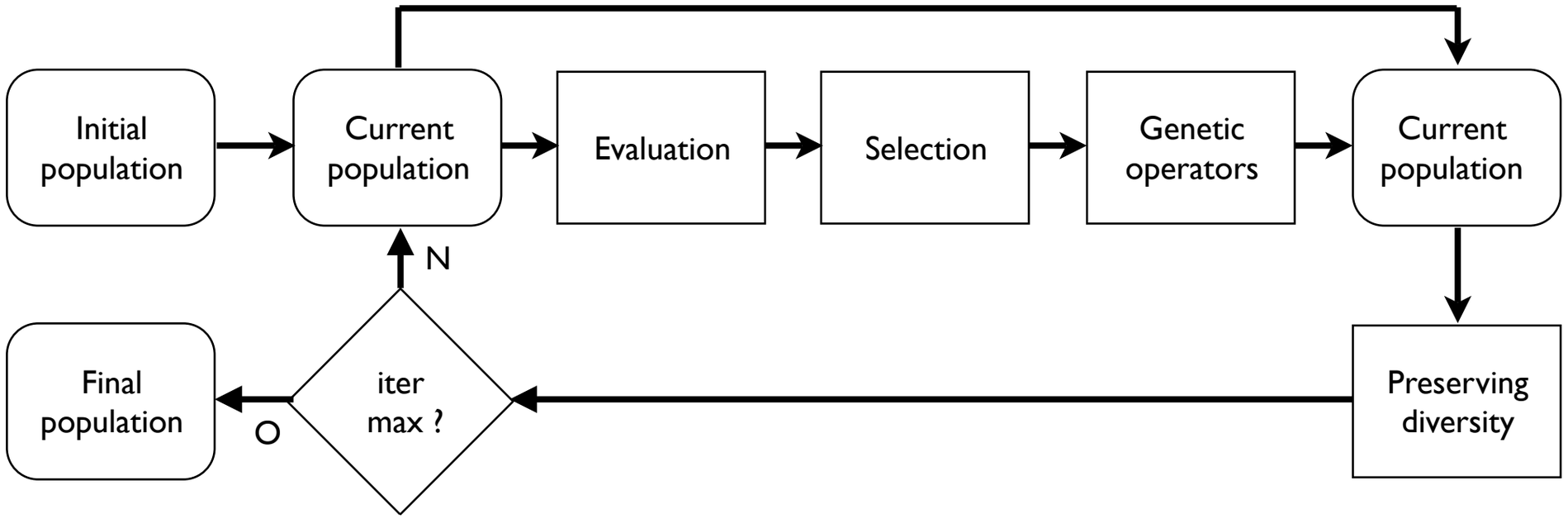
- Genetic algorithms are not well suited for constrained problems

Combining *orchidée* and *cdcsolver*

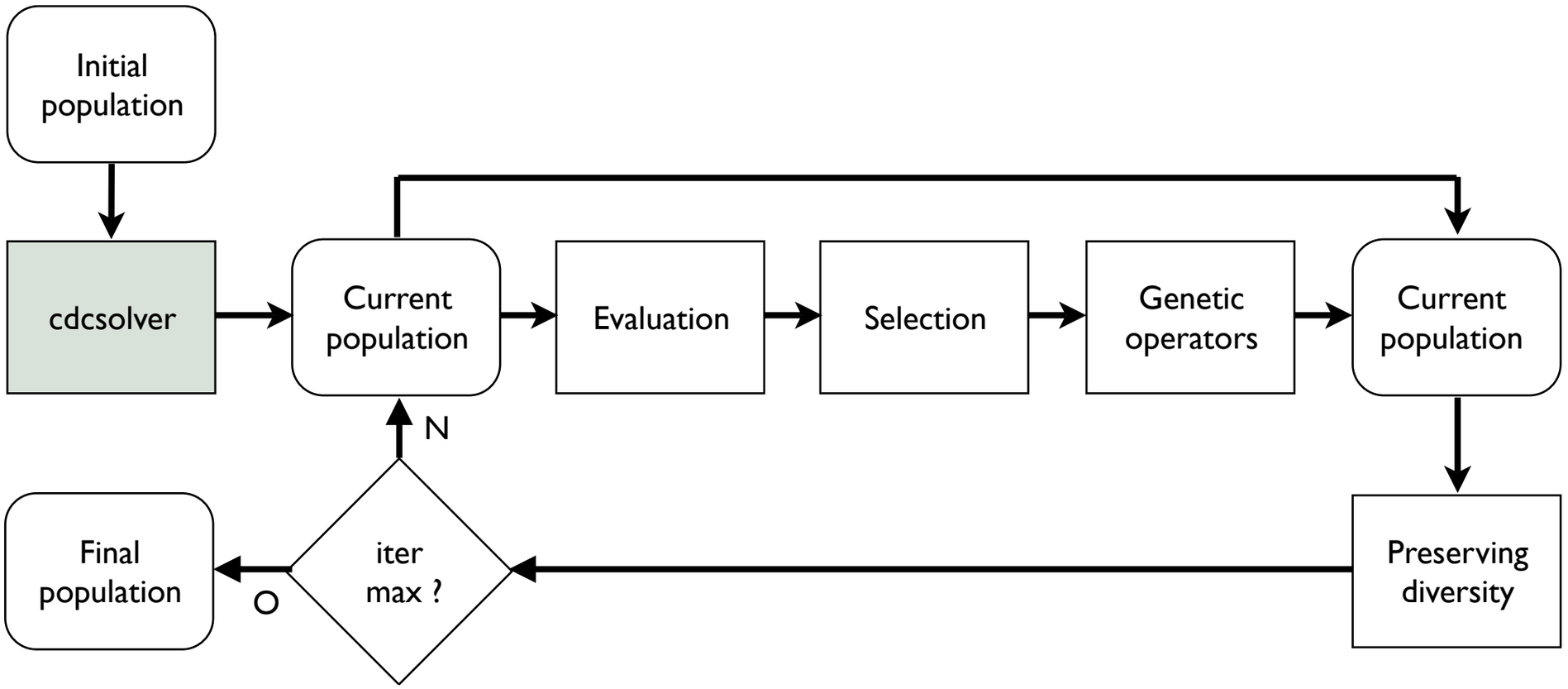
- Genetic algorithms are not well suited for constrained problems
- Repairing every inconsistent configuration is inefficient

Combining *orchidée* and *cdcsolver*

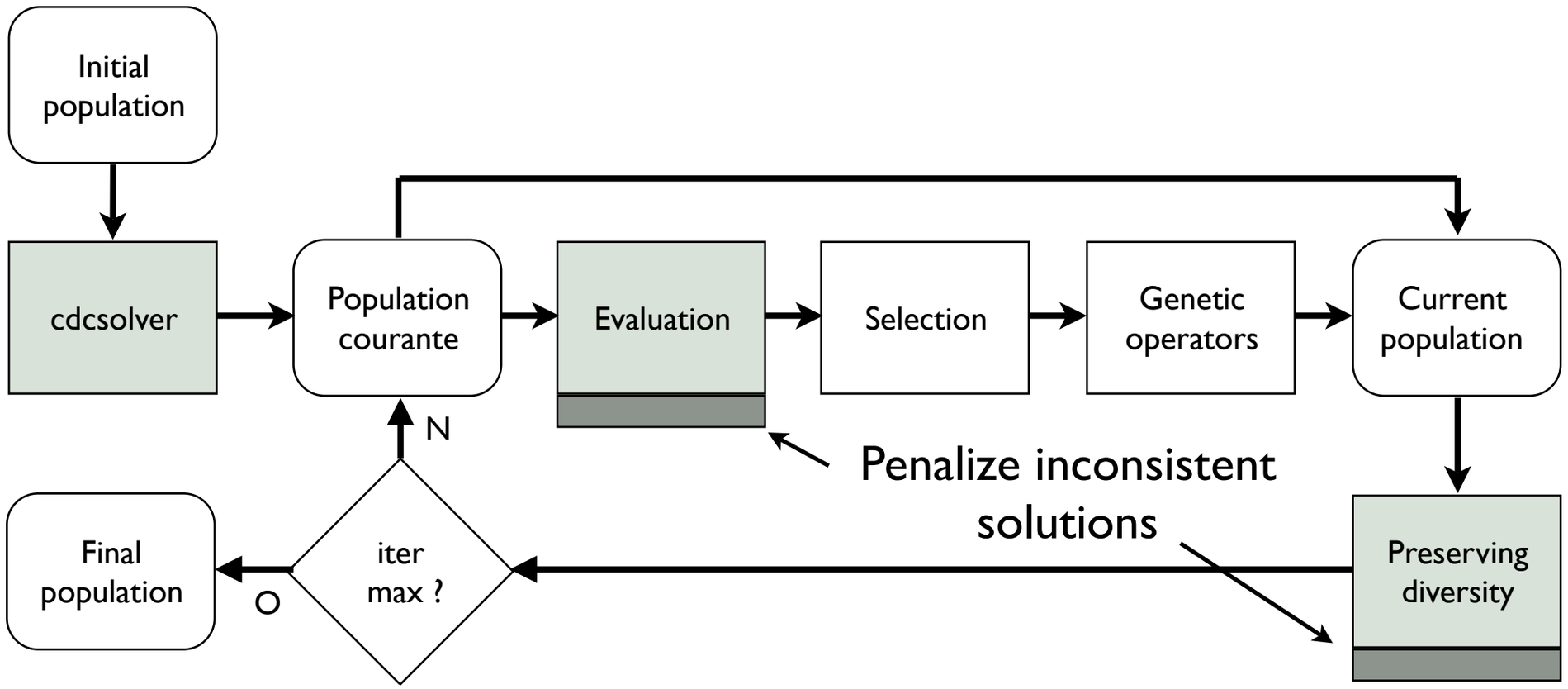
- Genetic algorithms are not well suited for constrained problems
- Repairing every inconsistent configuration is inefficient



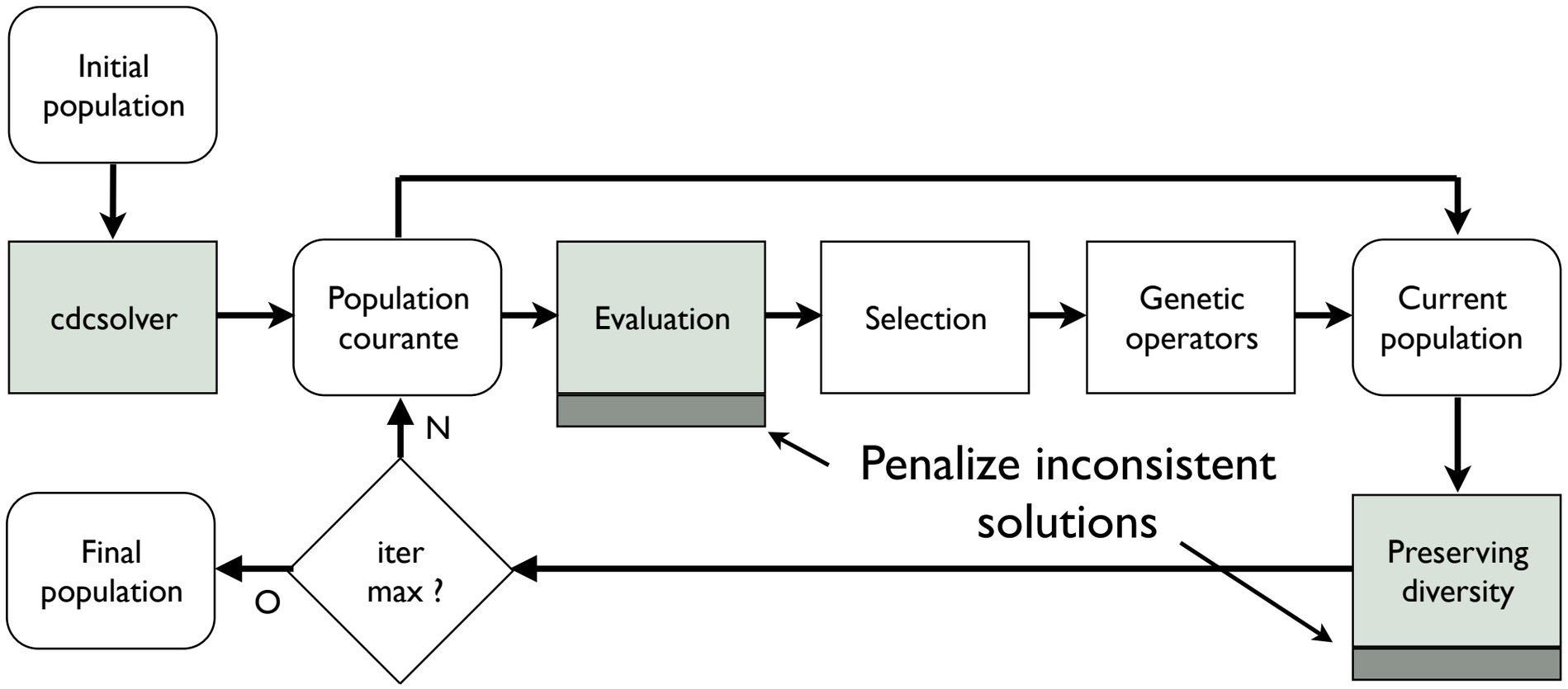
Combining *orchidée* and *cdcsolver*



Combining *orchidée* and *cdcsolver*

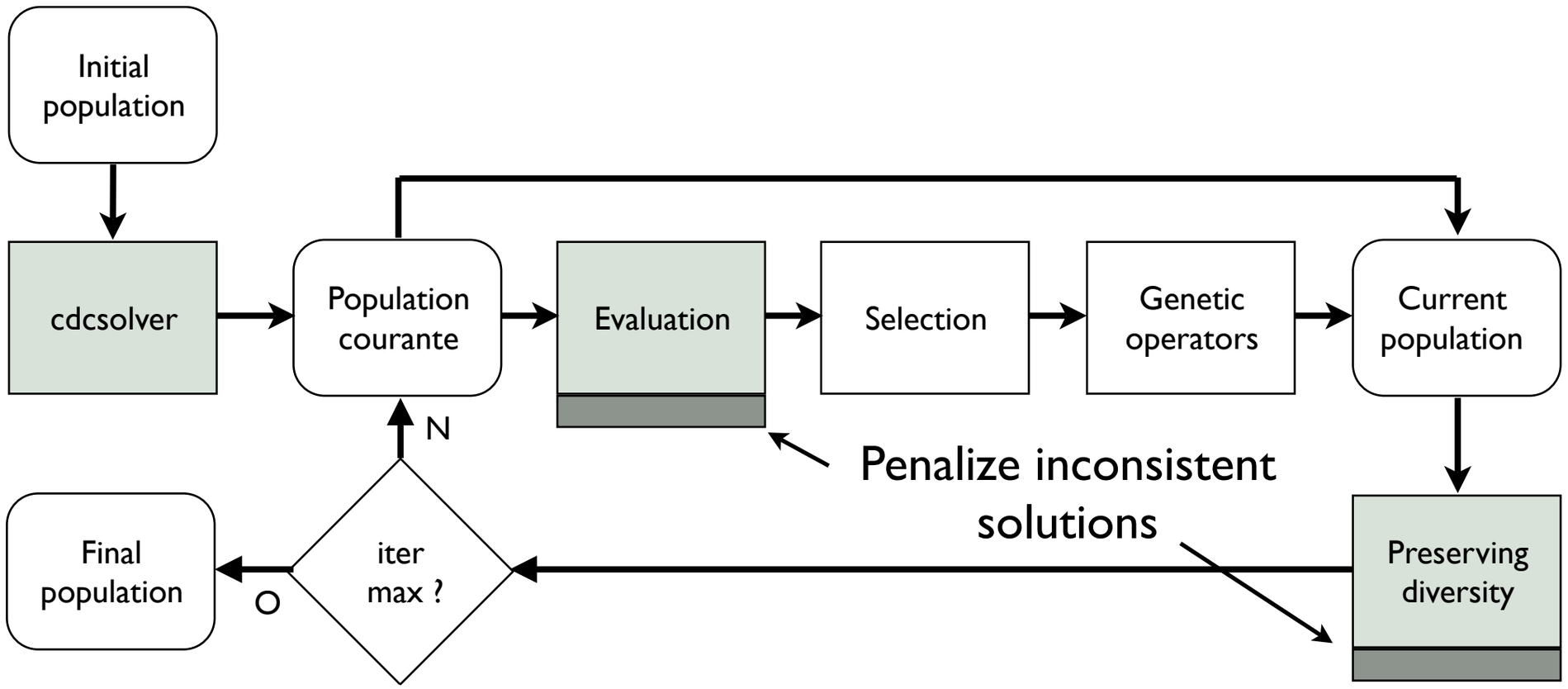


Combining *orchidée* and *cdcsolver*



Configurations comparing rules

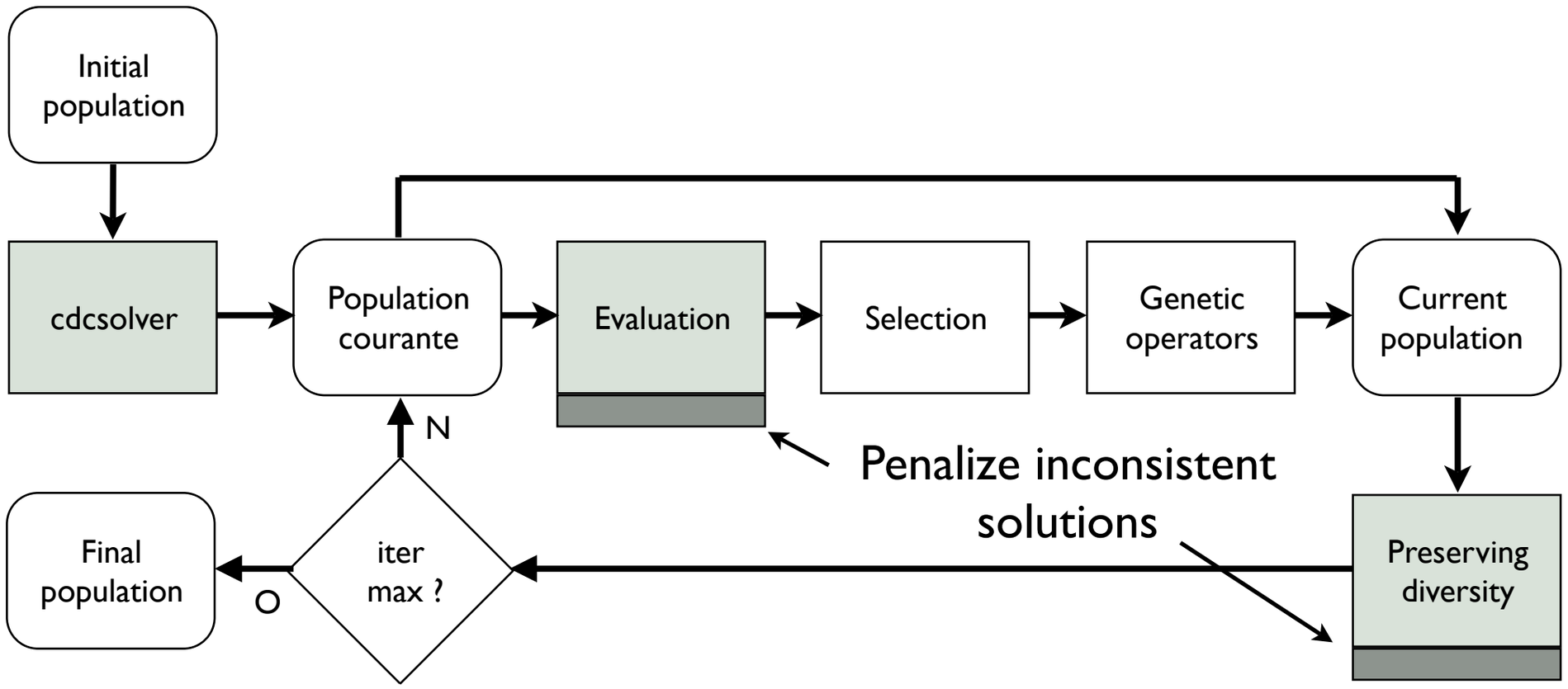
Combining *orchidée* and *cdcsolver*



Configurations comparing rules

1. Choose the more consistent configuration

Combining *orchidée* and *cdcsolver*



Configurations comparing rules

1. Choose the more consistent configuration
2. Choose the configuration with highest fitness according to the current Chebychev norm

Contents

Prototype

Prototype

- MATLAB code
- OpenSoundControl (OSC) interface [Wright&al.2003]
- Communicates with OpenMusic and Max/MSP

Prototype

- MATLAB code
- OpenSoundControl (OSC) interface [Wright&al.2003]
- Communicates with OpenMusic and Max/MSP

- Sound target analysis (feature extraction)
- Constrained multiobjective orchestration search
- User interaction - Listening preferences inference mechanism
- Enhancing timbre exploration with multiple views of solution set
- Manual editing / Auto-repairing and auto-transforming procedures

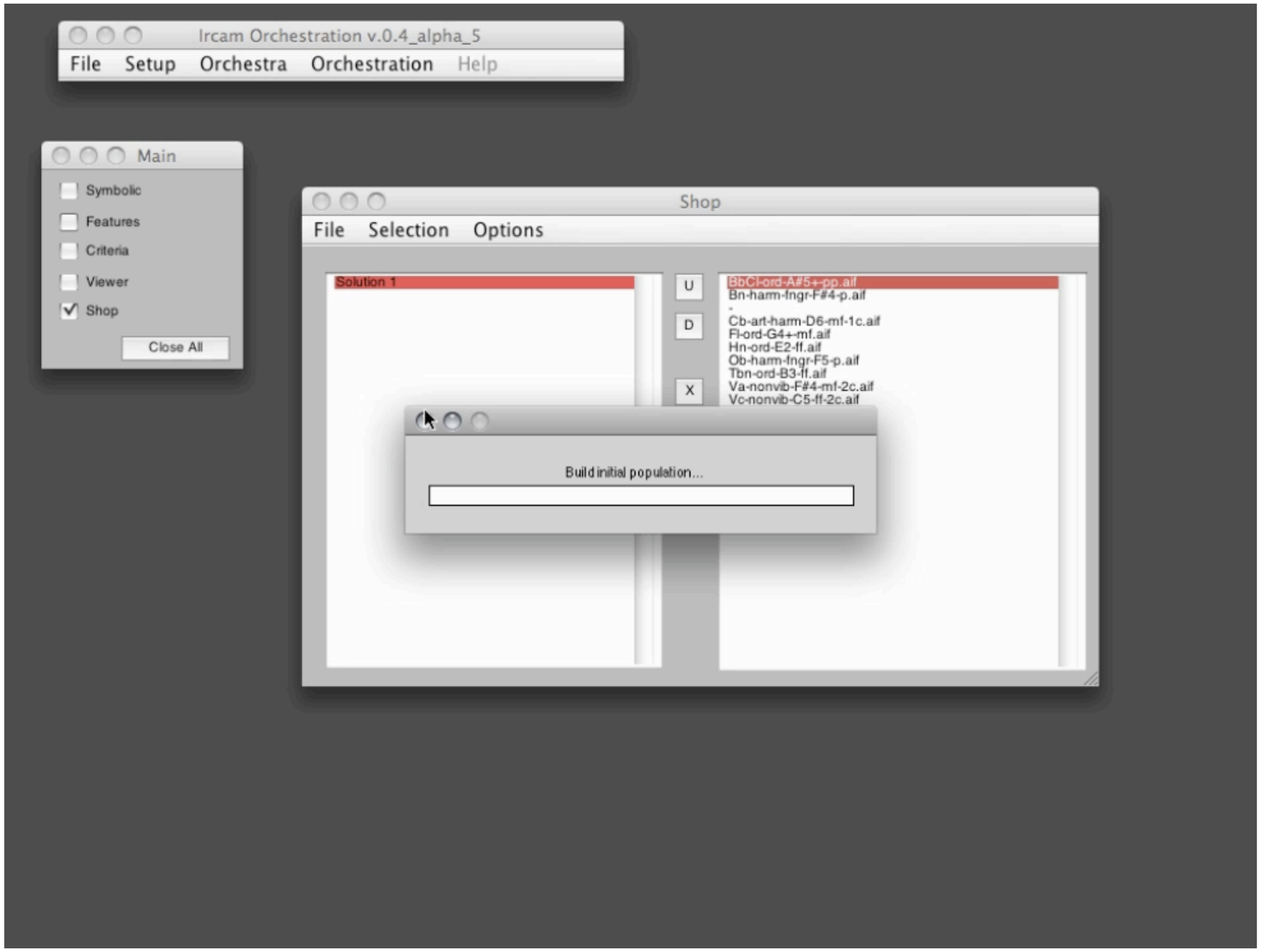
Prototype

- MATLAB code
- OpenSoundControl (OSC) interface [Wright&al.2003]
- Communicates with OpenMusic and Max/MSP

- Sound target analysis (feature extraction)
- Constrained multiobjective orchestration search
- User interaction - Listening preferences inference mechanism
- Enhancing timbre exploration with multiple views of solution set
- Manual editing / Auto-repairing and auto-transforming procedures

- Currently used by composers at IRCAM
- Standalone application packaging in progress

Demo


Ircam Orchestration v.0.4_alpha_5
File Setup Orchestra Orchestration Help
Main
Symbolic
Features
Criteria
Viewer
Shop
Close All
Shop
File Selection Options
Solution 1
U
D
X
BbCl-ord-A#5+-pp.aif
Bn-harm-fngr-F#4-p.aif
Cb-art-harm-D6-mf-1c.aif
Fl-ord-G4+-mf.aif
Hn-ord-E2-ff.aif
Ob-harm-fngr-F5-p.aif
Tbn-ord-B3-ff.aif
Va-nonvib-F#4-mf-2c.aif
Vc-nonvib-C5-ff-2c.aif
Build initial population...

More examples

- Car horn
- Tibetan horn
- No pre-recorded sound?
- Dynamic target
- Writing electronics

Contents

Conclusions

Conclusions

- Multiobjective combinatorial optimization model for the discovery of efficient solutions that approach a timbre target with a combination of instrument sounds

Conclusions

- Multiobjective combinatorial optimization model for the discovery of efficient solutions that approach a timbre target with a combination of instrument sounds
- Local search symbolic constraint solver that addresses compositional context issues

Conclusions

- Multiobjective combinatorial optimization model for the discovery of efficient solutions that approach a timbre target with a combination of instrument sounds
- Local search symbolic constraint solver that addresses compositional context issues
- Collaborative strategy for combining both methods in a single search process that handles potentially conflicting objectives

Conclusions

- Multiobjective combinatorial optimization model for the discovery of efficient solutions that approach a timbre target with a combination of instrument sounds
- Local search symbolic constraint solver that addresses compositional context issues
- Collaborative strategy for combining both methods in a single search process that handles potentially conflicting objectives
- Operational orchestration prototype already used in real-world musical situations

Current limitations

Current limitations

- Instrumental knowledge

Current limitations

- Instrumental knowledge
- Timbre description

Current limitations

- Instrumental knowledge
- Timbre description
- No model for unisons

Current limitations

- Instrumental knowledge
- Timbre description
- No model for unisons
- Some playing styles cannot be handled

Current limitations

- Instrumental knowledge
- Timbre description
- No model for unisons
- Some playing styles cannot be handled
- Global constraints only

Anyway ...

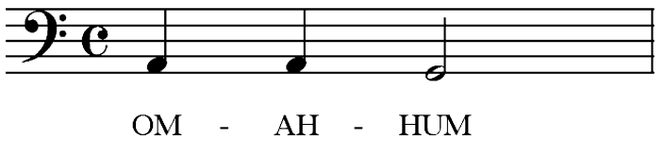
Anyway ...

Speakings (Jonathan Harvey)
[2008] For orchestra and electronics

Anyway ...

Speakings (Jonathan Harvey)
[2008] For orchestra and electronics

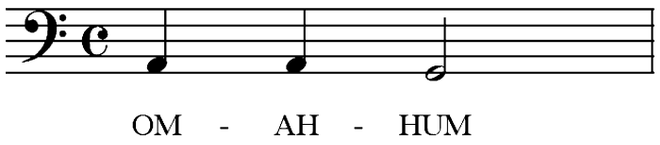
Harmonic background line
(beginning of the 3rd part)



Anyway ...

Speakings (Jonathan Harvey)
[2008] For orchestra and electronics

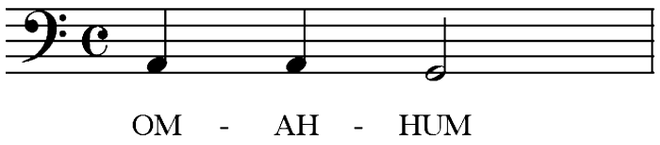
Harmonic background line
(beginning of the 3rd part)



Anyway ...

Speakings (Jonathan Harvey)
[2008] For orchestra and electronics

Harmonic background line
(beginning of the 3rd part)



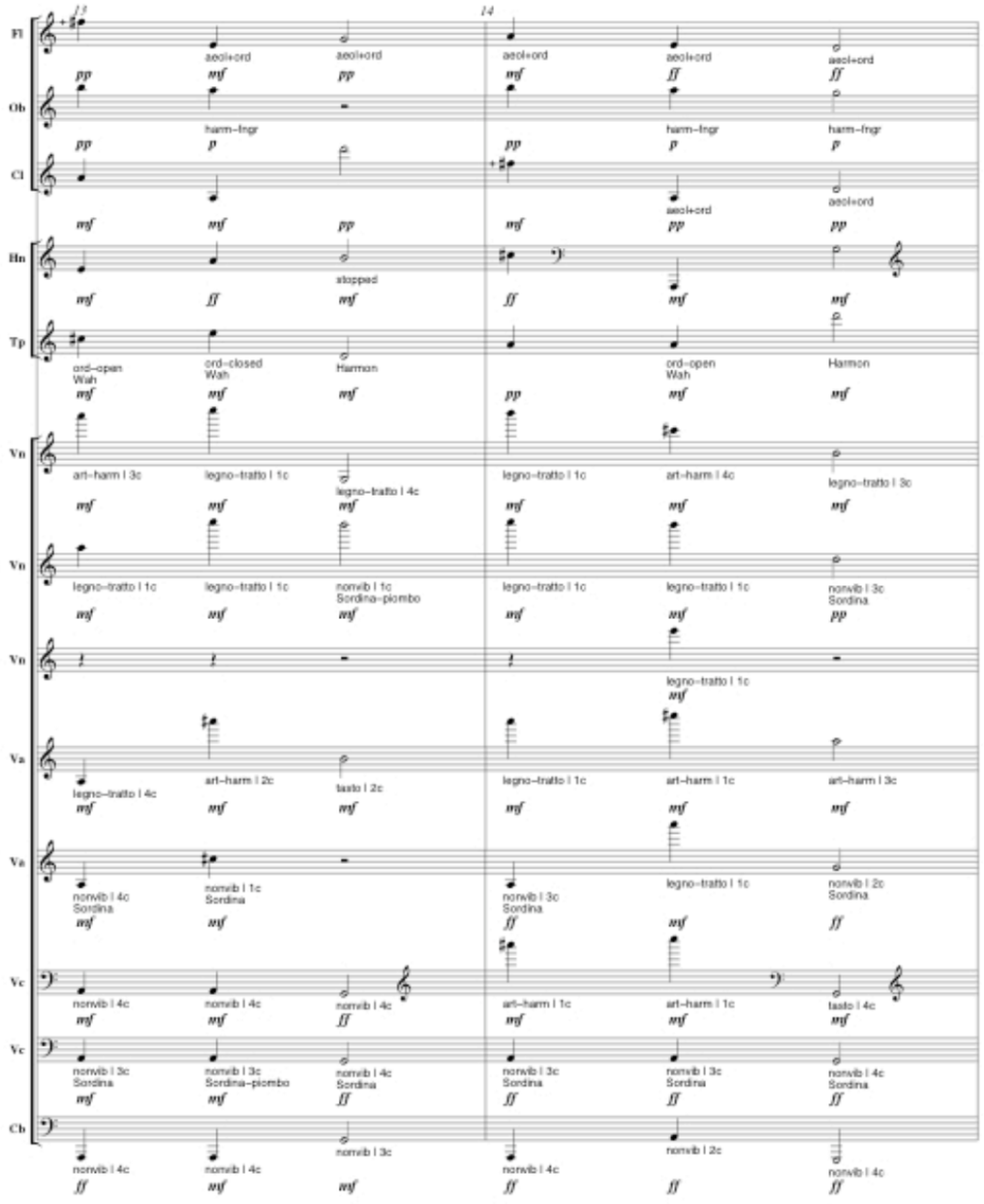
Orchestration :
- 22 ostinato repetitions
- Temporal evolution controlled by constraints

Anyway ...

Speakings (Jonathan Harvey)
[2008] For orchestra and electronics

Harmonic background line
(beginning of the 3rd part)

Orchestration :
- 22 ostinato repetitions
- Temporal evolution controlled by constraints

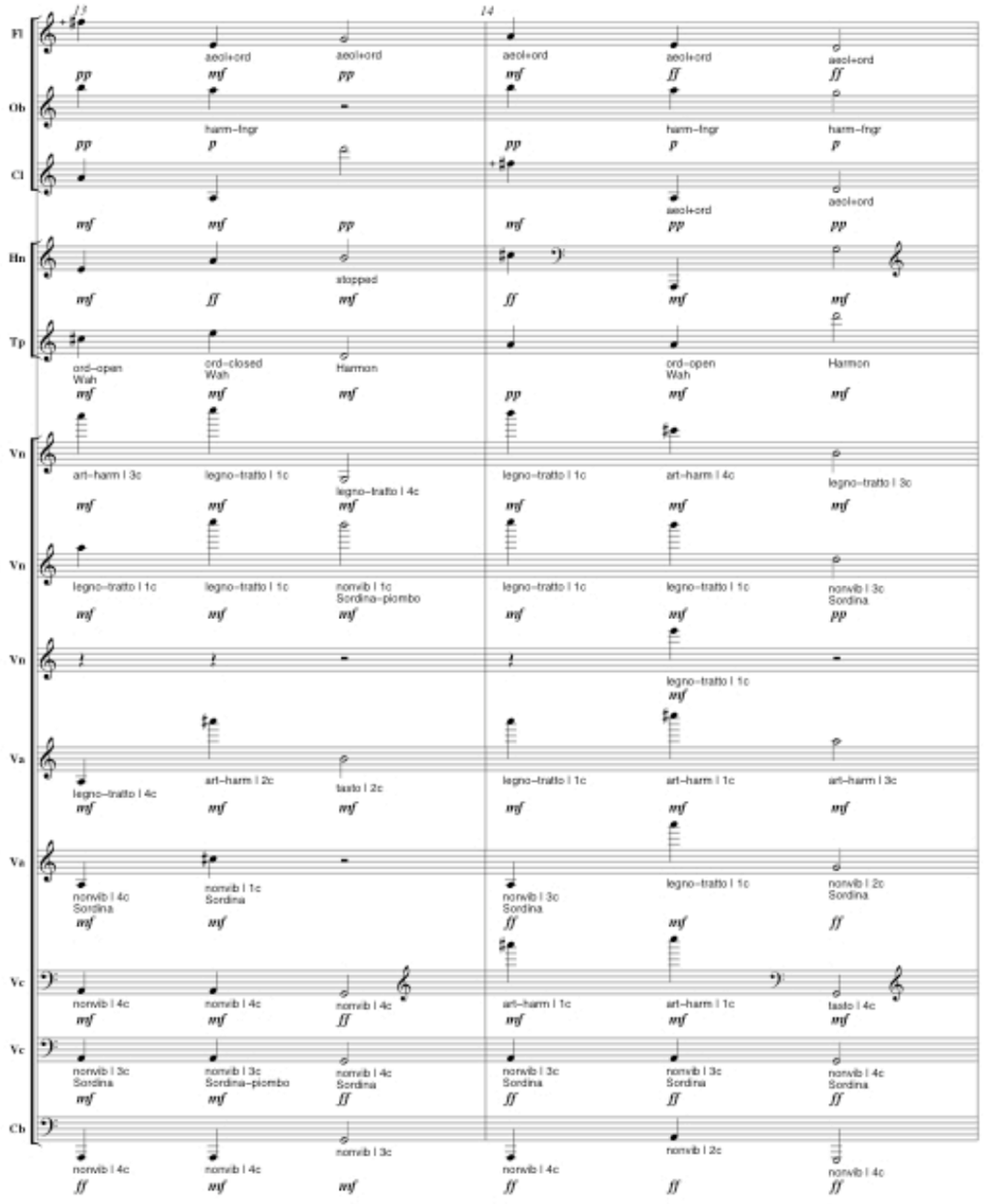


Anyway ...

Speakings (Jonathan Harvey)
[2008] For orchestra and electronics

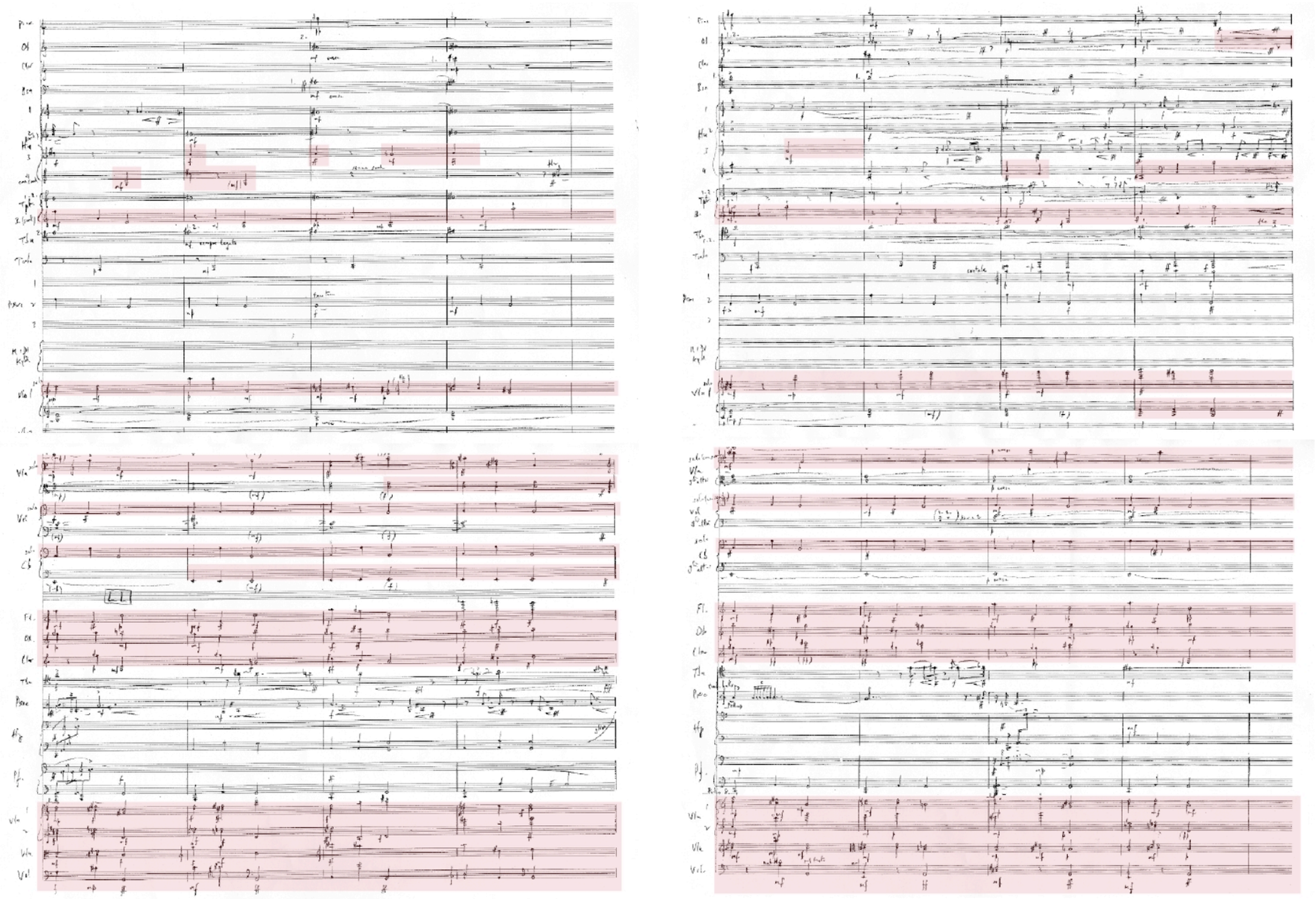
Harmonic background line
(beginning of the 3rd part)

Orchestration :
- 22 ostinato repetitions
- Temporal evolution controlled by constraints



Anyway ...

Speakings (Jonathan Harvey) - Created August 19th, 2008 in Royal Albert Hall, London (BBC Scottish Orchestra, director Ilan Volkov)



Future work

Future work

- Automatic criteria inference for high dimension problems

Future work

- Automatic criteria inference for high dimension problems
- Automatic constraint inference from target analysis

Future work

- Automatic criteria inference for high dimension problems
- Automatic constraint inference from target analysis
- Attack / sustain mechanisms

Future work

- Automatic criteria inference for high dimension problems
- Automatic constraint inference from target analysis
- Attack / sustain mechanisms
- Emergence (and disappearance)

Future work

- Automatic criteria inference for high dimension problems
- Automatic constraint inference from target analysis
- Attack / sustain mechanisms
- Emergence (and disappearance)
- Temporal model for “segmented” targets

Future work

- Automatic criteria inference for high dimension problems
- Automatic constraint inference from target analysis
- Attack / sustain mechanisms
- Emergence (and disappearance)
- Temporal model for “segmented” targets
- Temporal model for “articulated” targets

thank you