Process Mining using the Theory of Regions

Theory of Regions (for Languages)

The goal of language-based Theory of Regions is to synthesize a Petri net from a language defined by a collection of words, such that:

1) Each character corresponds to a transition in the Petri net (and vice-versa),
2) Each word in the language is an enabled trace in the Petri net (the Petri net is non-restrictive), and
3) Each enabled trace in the Petri net is a word in the language (the Petri net is minimal).

Theory of Regions (for Languages)

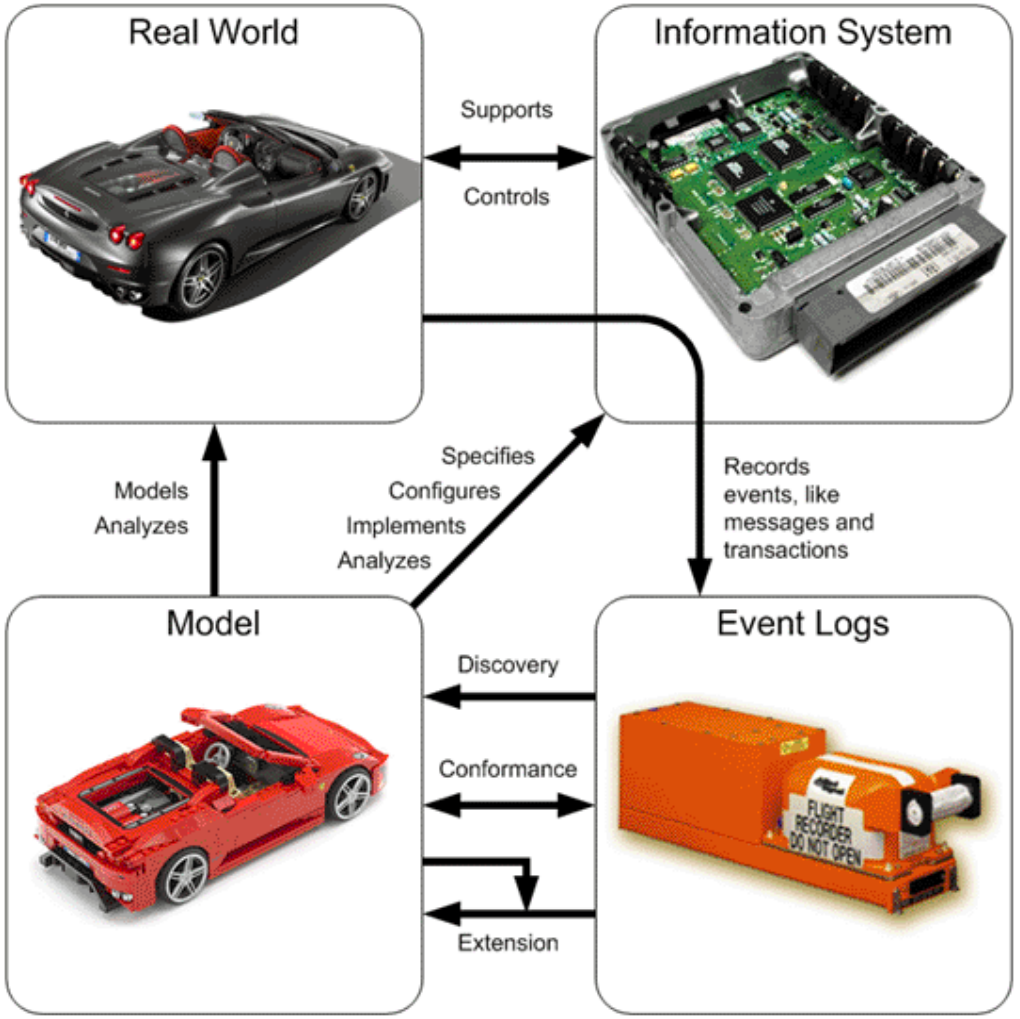
Algorithms exist for languages expressed as:

- (possibly infinite) sets of finite traces,
- (possibly infinite) sets of finite step executions, and
- (possibly infinite) set of labelled partial orders,

where the resulting Petri net is either:

- a p/t net,
- an elementary net, or
- an inhibitor net.

Process Mining: an overview



Log Files

Information systems typically log all kinds of events. Our basic assumption is that the log contains information about:

- specific *tasks* executed for
- specific *process instances* (cases, event-lists, audit trails).

Any knowledge of the underlying process is *not* assumed.

Process Mining VS. Theory of Regions

Process Mining

- **Event logs**
  finite sets of finite traces over events

- **Completeness unknown**
  Completeness of information is very unlikely.

- **Abstract representation required**

Theory of Regions

- **Languages**
  (possible infinite) sets of finite traces
  (or LPO's) over characters

- **Complete information provided**
  Completeness of information is guaranteed by the given language.

- **Exact and compact representation required**

How does it work?

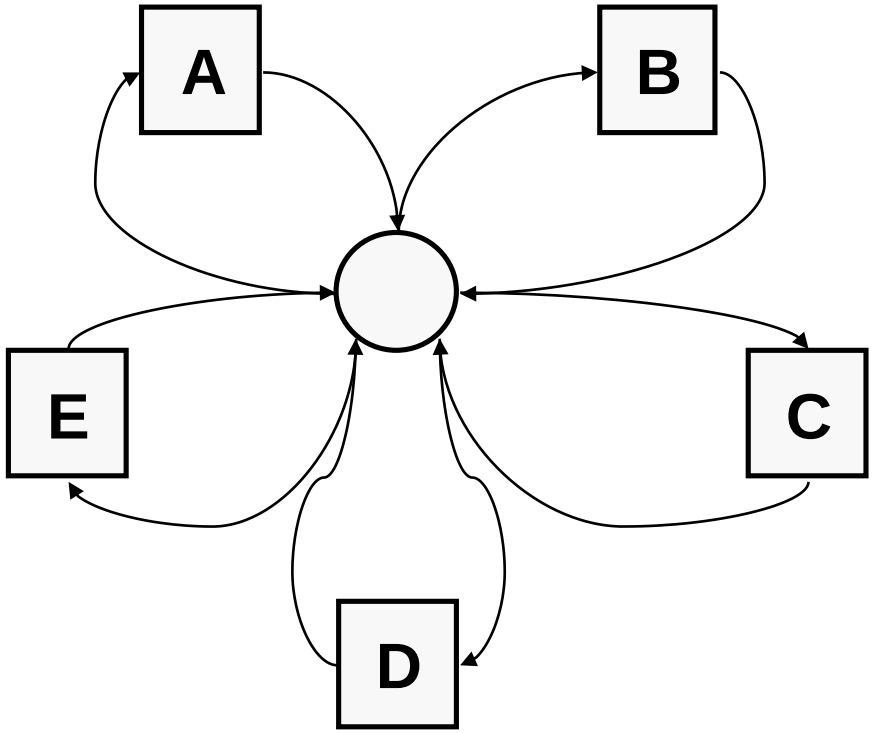
Consider the following traces:

abe^{10}, $abbe^{1}$, $acde^{100}$, $adce^{99}$

These traces lead to the following prefix-closed language:

a, ab, abe, abb, abbe,
ac, acd, acde,
ad, adc, adce

Transitions are fixed, so add places restricting the behavior.

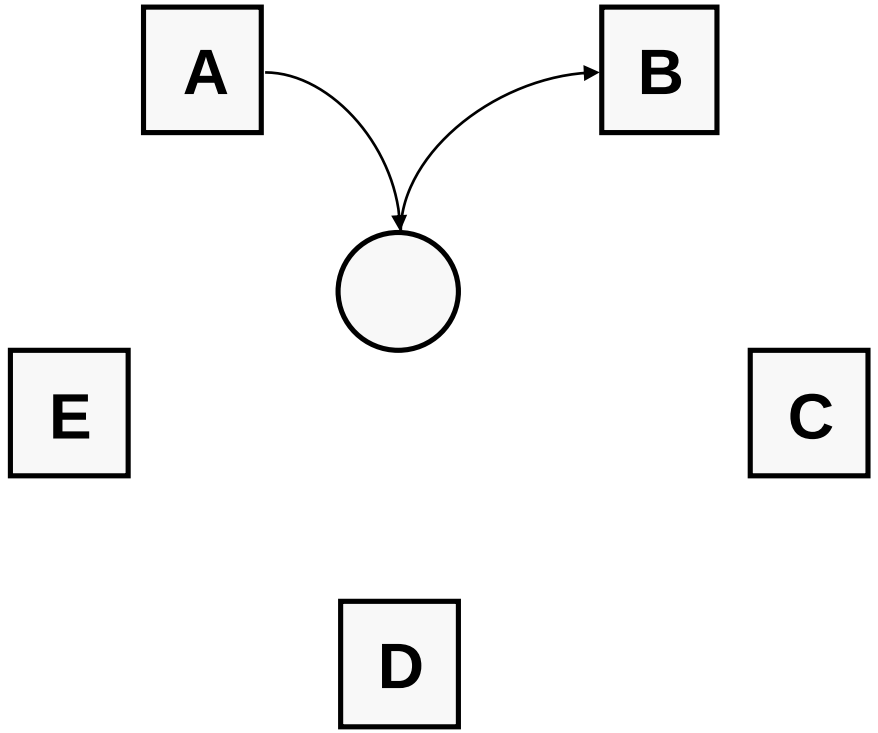


Candidate places

Is the place {A}->{B} a candidate?

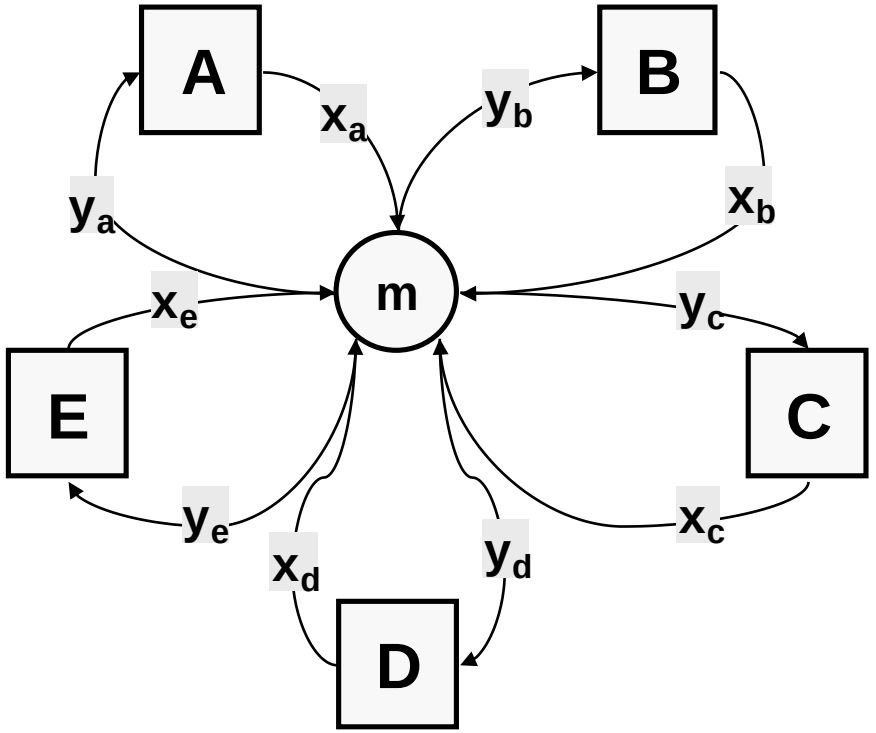
The place should be such that for all words in the prefix closure, enough tokens are produced by the prefix to execute the last transition:

[]a,
[a]b,
[ab]e,
[ab]b,
[abb]e,
[a]c,
[ac]d,
[acd]e,
[a]d,
[ad]c,
[adc]e



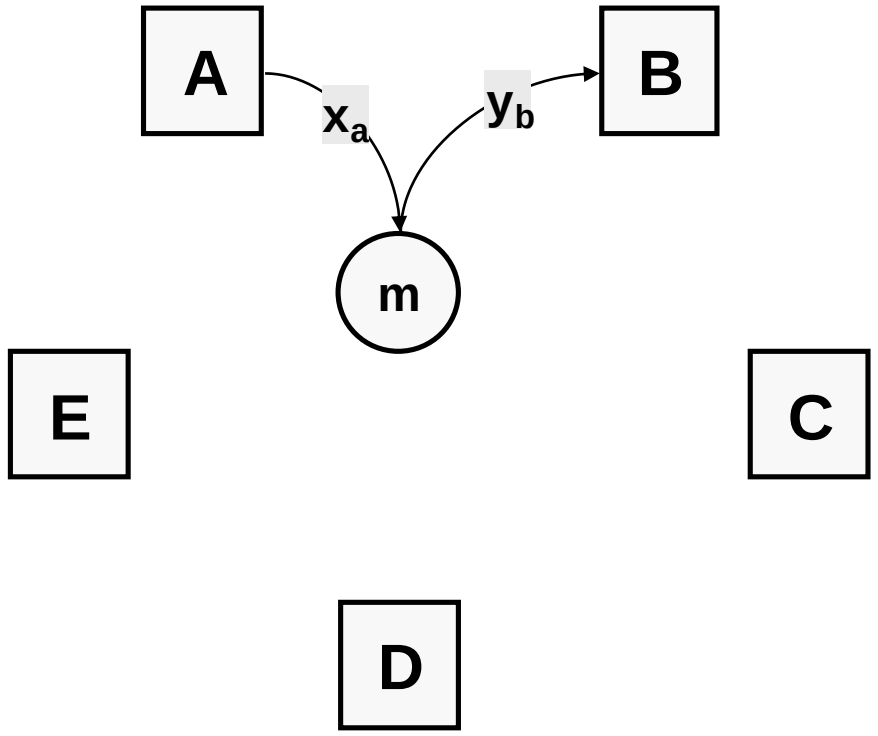
Integer linear programming

a	$m-y_a$	>=	0
ab	$m-y_a+x_a-y_b$	>=	0
abe	$m-y_a+x_a-y_b+x_b-y_e$	>=	0
abb	$m-y_a+x_a-y_b+x_b-y_b$	>=	0
abbe	$m-y_a+x_a-y_b+x_b-y_b+x_b-y_e$	>=	0
ac	$m-y_a+x_a-y_c$	>=	0
acd	$m-y_a+x_a-y_c+x_c-y_d$	>=	0
acde	$m-y_a+x_a-y_c+x_c-y_d+x_d-y_e$	>=	0
ad	$m-y_a+x_a-y_d$	>=	0
adc	$m-y_a+x_a-y_d+x_d-y_c$	>=	0
adce	$m-y_a+x_a-y_d+x_d-y_c+x_c-y_e$	>=	0



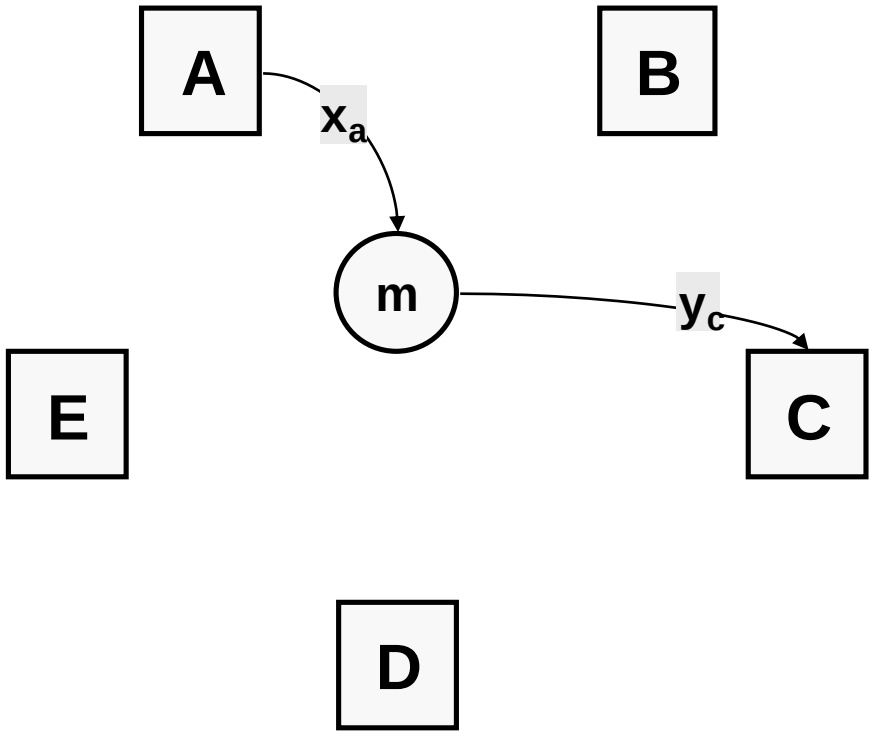
Integer linear programming

a	$m-y_a$	=	0
ab	$m-y_a+x_a-y_b$	=	0
abe	$m-y_a+x_a-y_b+x_b-y_e$	=	0
abb	$m-y_a+x_a-y_b+x_b-y_b$	=	-1
abbe	$m-y_a+x_a-y_b+x_b-y_b+x_b-y_e$	=	-1
ac	$m-y_a+x_a-y_c$	=	1
acd	$m-y_a+x_a-y_c+x_c-y_d$	=	1
acde	$m-y_a+x_a-y_c+x_c-y_d+x_d-y_e$	=	1
ad	$m-y_a+x_a-y_d$	=	1
adc	$m-y_a+x_a-y_d+x_d-y_c$	=	1
adce	$m-y_a+x_a-y_d+x_d-y_c+x_c-y_e$	=	1



Integer linear programming

a	$m-y_a$	=	0
ab	$m-y_a+x_a-y_b$	=	1
abb	$m-y_a+x_a-y_b+x_b-y_e$	=	1
abb	$m-y_a+x_a-y_b+x_b-y_b$	=	1
abbe	$m-y_a+x_a-y_b+x_b-y_b+x_b-y_e$	=	1
ac	$m-y_a+x_a-y_c$	=	0
acd	$m-y_a+x_a-y_c+x_c-y_d$	=	0
acde	$m-y_a+x_a-y_c+x_c-y_d+x_d-y_e$	=	0
ad	$m-y_a+x_a-y_d$	=	1
adc	$m-y_a+x_a-y_d+x_d-y_c$	=	0
adce	$m-y_a+x_a-y_d+x_d-y_c+x_c-y_e$	=	0



Two alternatives for Region Theory

Basis representation

Calculate a basis of integral vectors, such that every solution of the given system of linear inequations is a linear combination of these vectors.

our example:

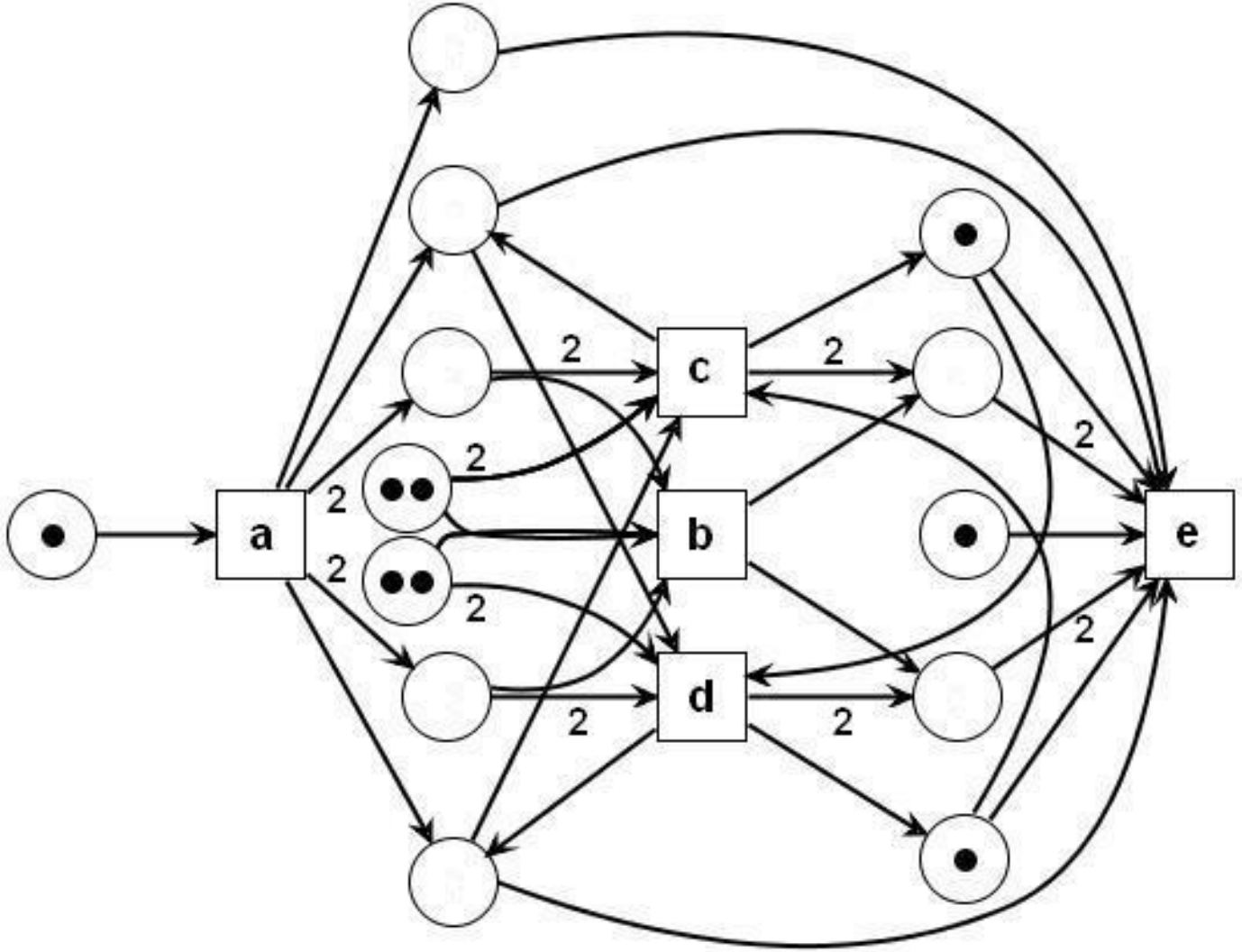
55 places, of which 50 redundant

Separating representation

Calculate a set of integral vectors, such that no wrong (unobserved) continuations are possible for any prefix.

our example:

6 places, of which 1 redundant


a
b
c
d
e
2
2
2
2
2
2
2
2
2
2
2

Problems with region theory

Let *L* be an event log over a set of activities *T*, such that
- there are $|T| = 20$ different activities, and
- the log contains $|L| = 1.000.000$ events.

The theoretical upper bound for any resulting net, is such that is contains:
- $|T| = 20$ transitions, and
- *exponential(* $|L|$ *)* $>>> 1x10^{12}$ *places* (basis representation) or
- $|T| * |L| = 2x10^{7}$ *places* (separating representation).

i.e. the number of places scales in the size of the log (i.e. the number of cases)

Region theory in Process Mining

In process mining the aim is to obtain a compact abstract representation

Only include those places that are somehow “promising”

1. Include a target function in the constraints to exploit causalities
2. Ensure relaxed soundness of the resulting model
3. Mitigate the effect of infrequent behavior’

How does it work?

Consider the following traces:

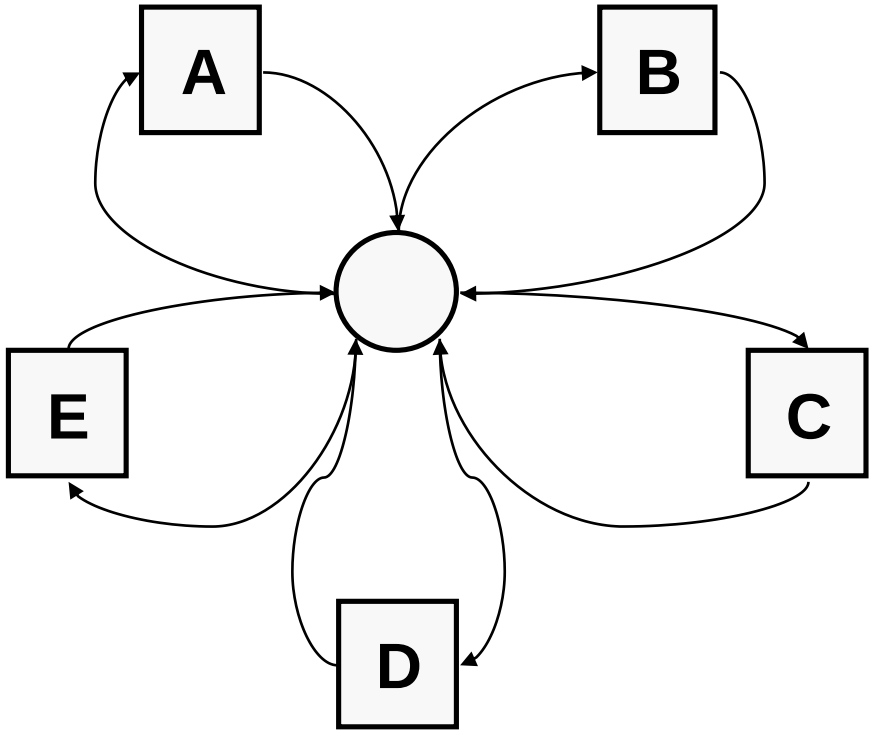
abbe, acde, adce

These traces lead to the following causal dependencies:

a → b, b → b, b → e,
a → c, d → e,

a → d, c → e,

Add places restricting the behavior, but expressing the causal dependencies.



Add a target function

Add a target function to the ILP, i.e. Minimize $c_m m + c_x x + c_y y$

Where c_m and vectors c_x and c_y are coefficients indicating a preference to add specific arcs.

Minimize incoming arcs and maximize outgoing arcs per place

Werf, J.M.E.M. van der, Dongen, B.F. van, Hurkens, C.A.J., Serebrenik, A.: Process Discovery using Integer Linear Programming. Fundam. Inform. 94(3-4), 387–412 (2009). DOI 10.3233/FI-2009-136. URL

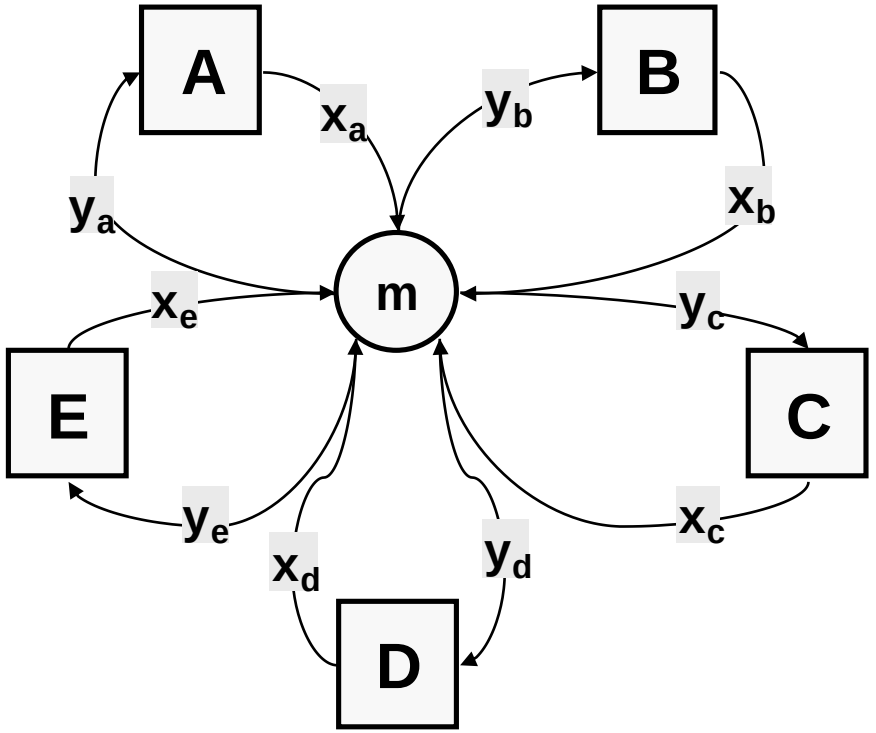
Minimize the time a token spends in a place

Favor minimal regions

Zelst, S.J. van, Dongen, B.F. van, Aalst, W.M.P. van der: ILP-Based Process Discovery Using Hybrid Regions. In: Aalst, W.M.P. van der, Bergenthum, R., Carmona, J. (ed.) Proceedings of the ATAED 2015 Workshop,

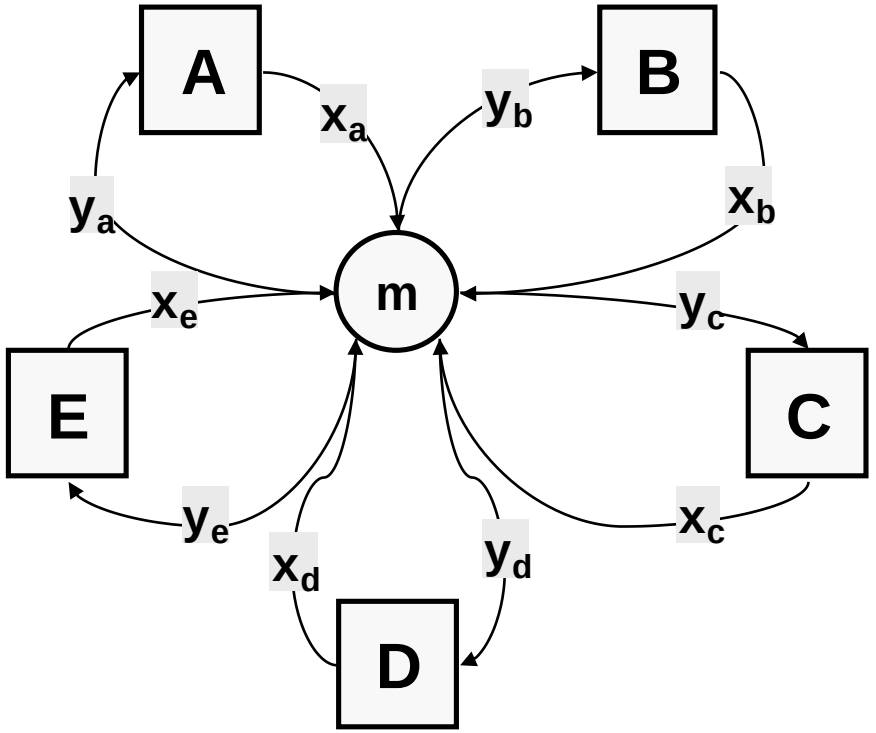
Exploit causalities

a	$m-y_a$	>=	0
ab	$m-y_a+x_a-y_b$	>=	0
	…		
ad	$m-y_a+x_a-y_d$	>=	0
adc	$m-y_a+x_a-y_d+x_d-y_c$	>=	0
adce	$m-y_a+x_a-y_d+x_d-y_c+x_c-y_e$	>=	0
init	$m = 0$		
a → b	$x_a = y_b = 1$		



Ensure relaxed sound results

a	$m-y_a$	>=	0
ab	$m-y_a+x_a-y_b$	>=	0
	…		
ad	$m-y_a+x_a-y_d$	>=	0
adc	$m-y_a+x_a-y_d+x_d-y_c$	>=	0
adce	$m-y_a+x_a-y_d+x_d-y_c+x_c-y_e$	>=	0
abbe	$m-y_a+x_a-y_b+x_b-y_b+x_b-y_e+x_e$	=	0
adce	$m-y_a+x_a-y_d+x_d-y_c+x_c-y_e+x_e$	=	0

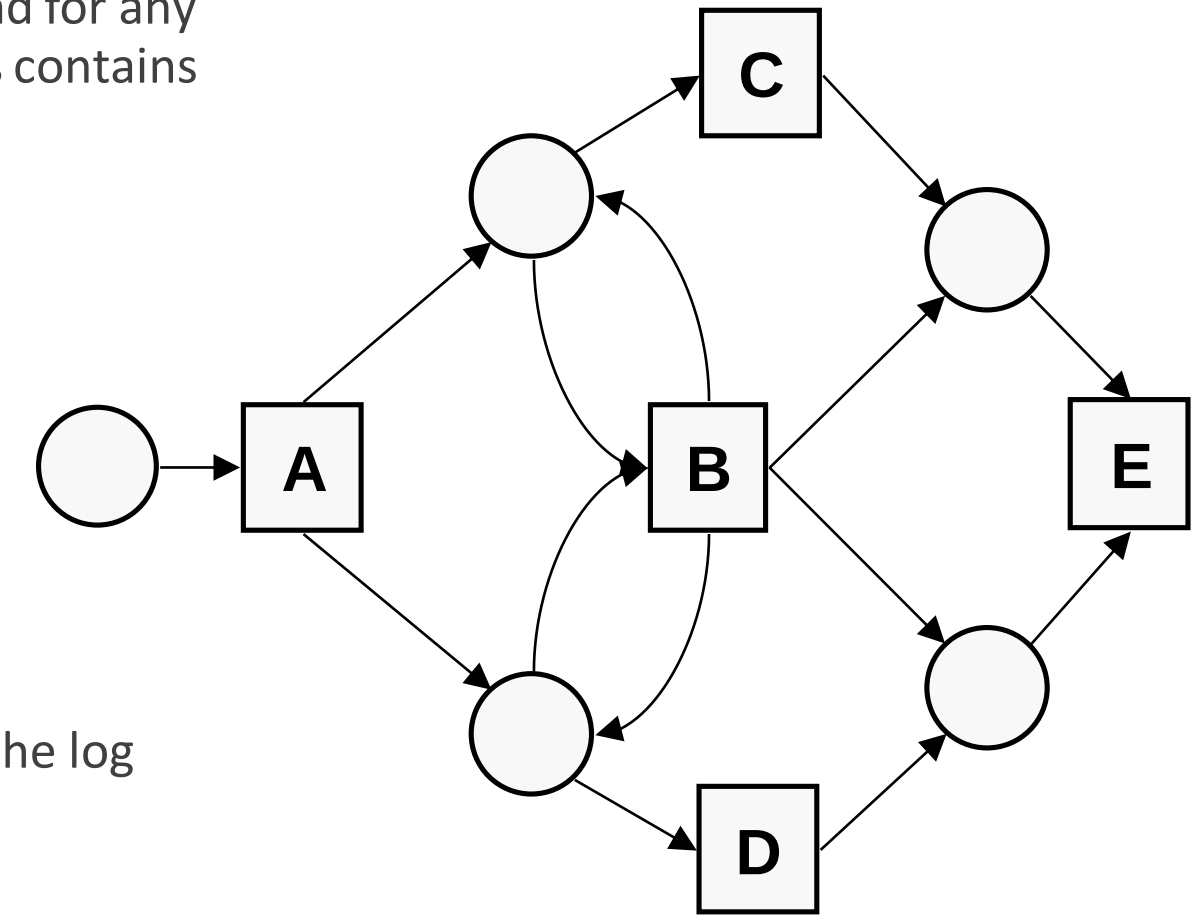


Result

The theoretical upper bound for any resulting net, is such that is contains at most:

- | *T* | = *20* transitions, and
- | *T* |* | *T* | = *400 places*

i.e. the number of places is *independent* of the size of the log



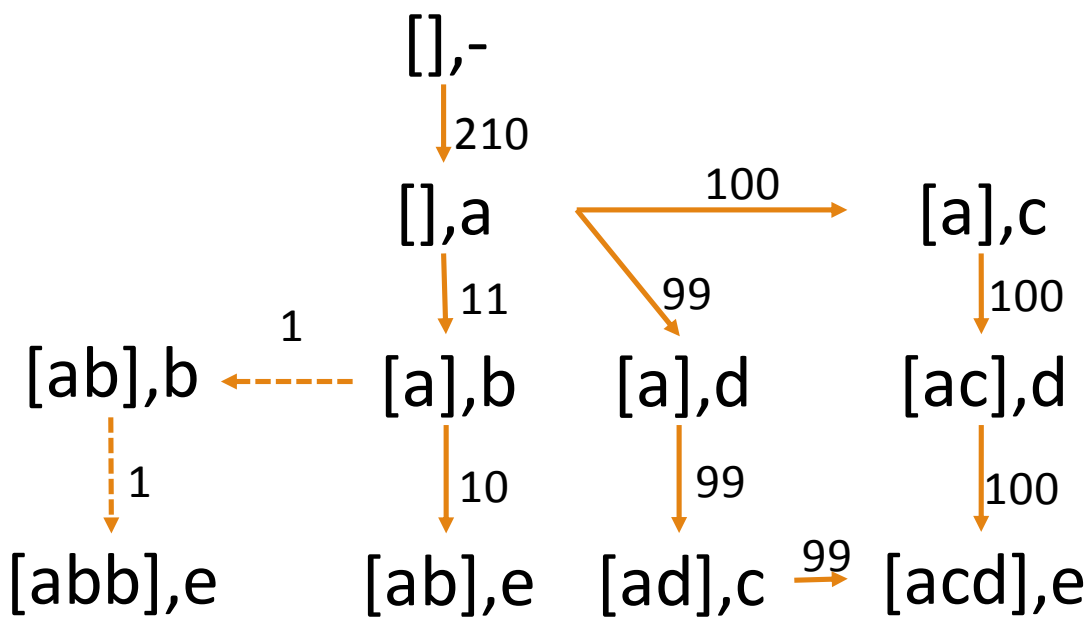
Dealing with infrequent behavior

Using a sequence encoding graph, we can prioritize the constraints that are more frequent in the event log

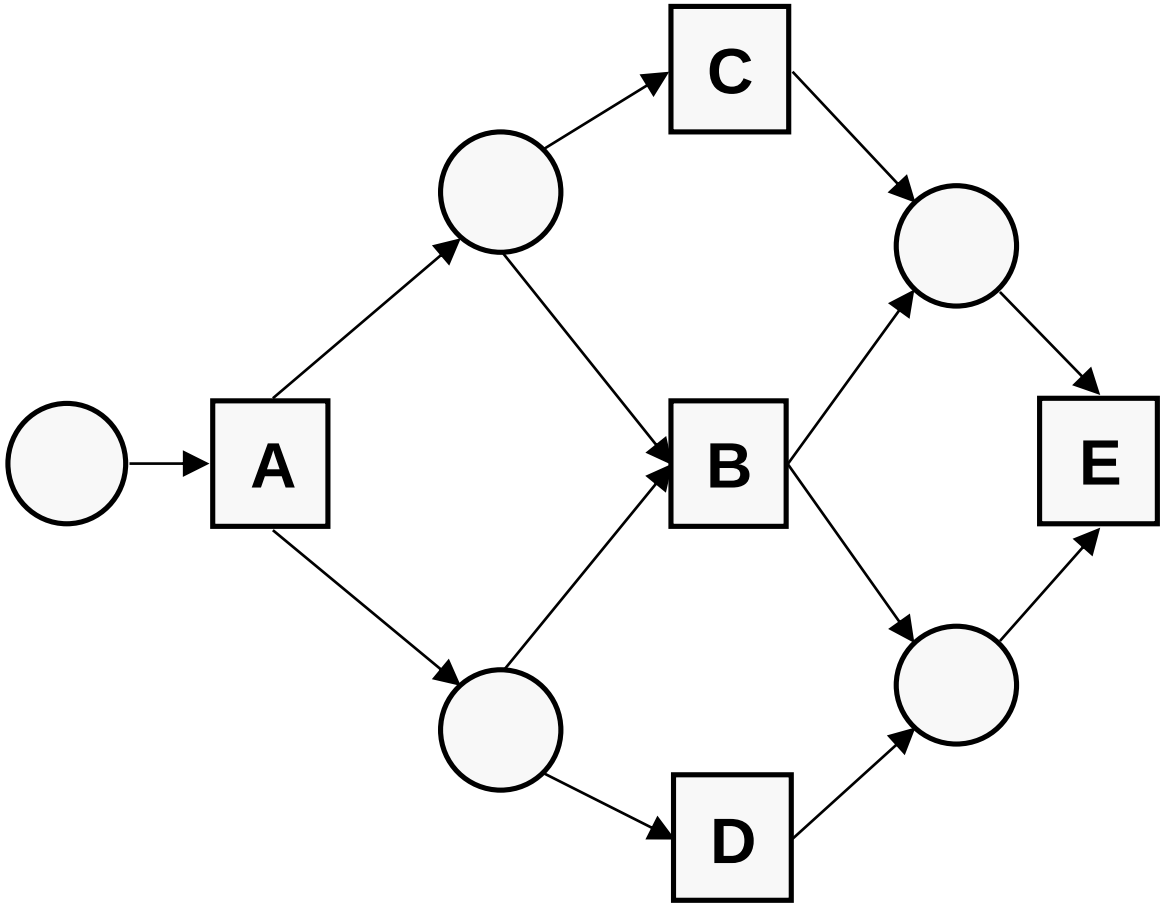
Sebastiaan J. van Zelst, Boudewijn F. van Dongen, Wil M. P. van der Aalst, H. M. W. Verbeek:
Discovering Relaxed Sound Workflow Nets using Integer Linear Programming.
CoRR abs/1703.06733 (2017)

Recall our example log:

abe^{10},
$abbe^{1}$,
$acde^{100}$,
$adce^{99}$



Result



Conclusions

Using our approach, the Theory of Regions can be applied in the context of process mining, in such a way that the size of the resulting Petri net is independent of the size of the log.

The resuling Petri net can be a regular P/T net, an extended free-choice P/T net, a state machine or a marked graph.

Downsides remain the completeness assumption and the resulting model, since this is not an abstraction of the log, which is often required in process mining.