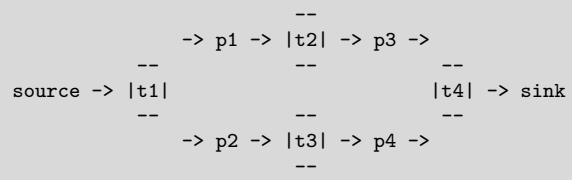
MODULE *Example3_Parallel*

Parallelize and synchronize flow through a Petri Net. This makes Petri Nets very useful for modelling distributed systems. In this example some properties include that both "t2" and "t3" may fire before the other and that "t4" can only fire after "t2" and "t3" have fired.



$Places \triangleq \{$"source", "p1", "p2", "p3", "p4", "sink"$\}$ Define the net.
$Transitions \triangleq \{$"t1", "t2", "t3", "t4"$\}$
$Arcs \triangleq [$
$source \mapsto \{$"t1"$\}$,
$p1 \mapsto \{$"t2"$\}$,
$p2 \mapsto \{$"t3"$\}$,
$p3 \mapsto \{$"t4"$\}$,
$p4 \mapsto \{$"t4"$\}$,

$t1 \mapsto \{$"p1", "p2"$\}$,
$t2 \mapsto \{$"p3"$\}$,
$t3 \mapsto \{$"p4"$\}$,
$t4 \mapsto \{$"sink"$\}$
$]$
$ArcWeights \triangleq \langle\rangle$ Unspecified arc weights default to 1.
$InitialMarking \triangleq [source \mapsto 1]$
VARIABLE *Marking*

$PN \triangleq$ INSTANCE *PetriNet* Instantiate it within a namespace.

$Spec \triangleq PN!Spec$ Make *Spec* and *Invariants* available for the config file.

$Invariants \triangleq PN!Invariants$

Properties

Eventually, we arrive as a expected final marking.
$FinalMarking \triangleq PN!FinalMarking([sink \mapsto 1])$

IsStateMachine property would not hold.

$IsFreeChoiceNet \triangleq PN!IsFreeChoiceNet$