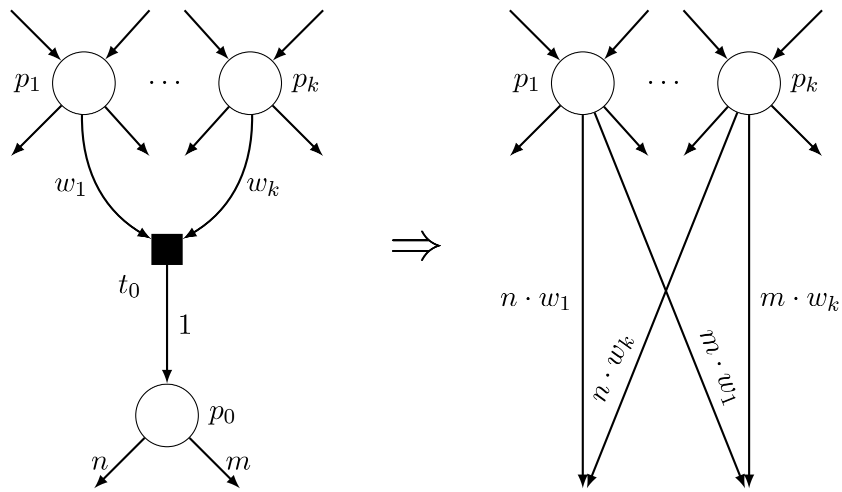
Rule A: Sequential transition removal

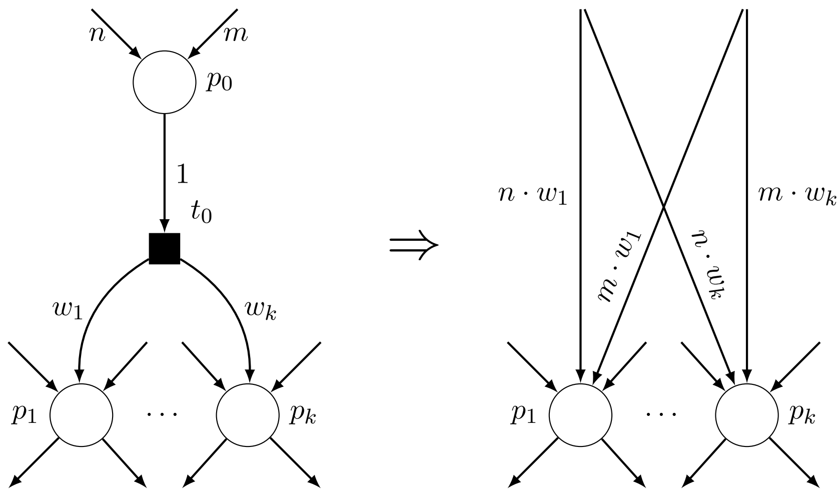
Rule A merges sequential transitions, i.e. a transition and another transition that must precede or follow it. Rule A is equivalent to a pre (or post) agglomeration with exactly one producer (or consumer) with a weight of 1. The two variants of Rule A can be seen in Figure 1 and Figure 2.

Theorem 1 *The two variants of Rule A in Figure 1 and Figure 2 are both correct for LTL$\backslash$X cardinality properties.*



Precondition	Update
Fix p_0 and t_0 where ${}^\bullet t_0 = \{p_1, \ldots, p_k\}$ s.t.: A1) $t_0^\bullet = \{p_0\}$ and $\boxplus(t_0, p_0) = 1$ A2) ${}^\bullet p_0 = \{t_0\}$ and $p_0 \notin \{p_1, \ldots, p_k\}$ A3) $p_0^\circ = p_1^\circ = \cdots = p_k^\circ = {}^\circ t_0 = \emptyset$ A4) $\{p_0, p_1, \ldots, p_k\} \cap places(\varphi) = \emptyset$ A5) $M_0(p_0) = 0$	UA1) For all $t \in p_0^\bullet$ and all $p \in \{p_1, \ldots, p_k\}$ set $\boxminus'(p, t) := \boxminus(p, t) + \boxminus(p_0, t) \cdot \boxminus(p, t_0)$ UA2) Remove p_0 and t_0

Figure 1: Rule A: Sequential transition removal (pre)



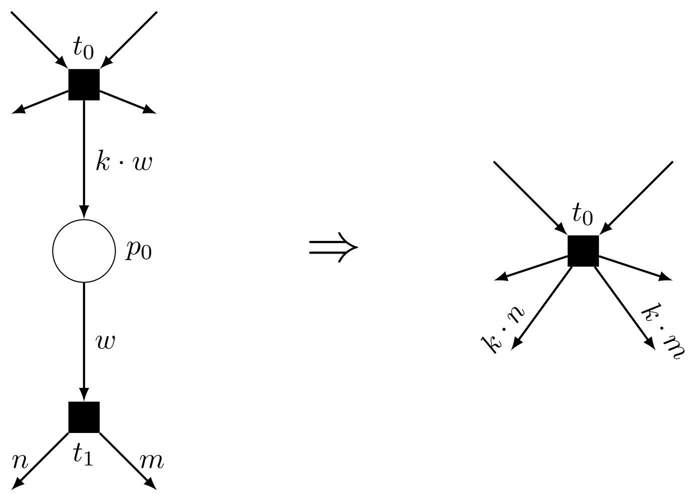
Precondition	Update
Fix p_0 and t_0 where $t_0^\bullet = \{p_1, \ldots, p_k\}$ s.t.: A1) $^\bullet t_0 = \{p_0\}$ and $\boxminus(p_0, t_0) = 1$ A2) $p_0^\bullet = \{t_0\}$ and $p_0 \notin \{p_1, \ldots, p_k\}$ A3) $p_0^\circ = p_1^\circ = \cdots = p_k^\circ = {}^\circ t_0 = \emptyset$ A4) $\{p_0, p_1, \ldots, p_k\} \cap places(\varphi) = \emptyset$	UA1) For all $p \in \{p_1, \ldots, p_k\}$ change the initial marking s.t. $M_0'(p) := M_0(p) + M_0(p_0) \cdot \boxplus(t_0, p)$ UA2) For all $t \in {}^\bullet p_0$ and all $p \in \{p_1, \ldots, p_k\}$ set $\boxplus'(t, p) := \boxplus(t, p) + \boxplus(t, p_0) \cdot \boxplus(t_0, p)$ UA3) Remove p_0 and t_0

Figure 2: Rule A: Sequential transition removal (post)

Rule B: Sequential place removal

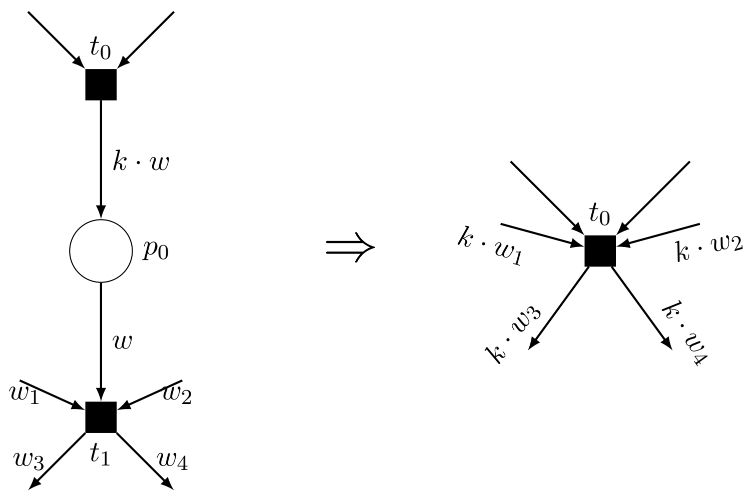
Rule B merges two transitions surrounding a place with no other transitions than the two. Rule B is equivalent to an agglomeration with exactly one producer and one consumer, but allow them to have different weights. Hence, there is a pre- and post-agglomeration variant of Rule B defined in Figure 3 and Figure 4, respectively.

Theorem 2 *The two variants of Rule B in Figure 3 and Figure 4 are both correct for LTL$\setminus$X cardinality properties.*



Precondition	Update
Fix p_0 and t_0, t_1 where $t_0 \neq t_1$ s.t.: B1) ${}^\bullet p_0 = \{t_0\}, p_0^\bullet = \{t_1\}, {}^\bullet t_1 = \{p_0\}$ B2) $\boxplus(t_0, p_0) = k \cdot \boxminus(p_0, t_1)$ for $k \geq 1$ B3) $p_0^\circ = {}^\circ t_0 = {}^\circ t_1 = \emptyset$ B4) $p_0 \notin places(\varphi)$ B5) $p^\circ = \emptyset$ and $p \notin places(\varphi)$ for all $p \in t_1^\bullet$	UB1) For all $p \in P \setminus \{p_0\}$ set $M_0'(p) := M_0(p) + \lfloor M_0(p_0) / \boxminus(p_0, t_1) \rfloor \cdot \boxplus(t_1, p)$ UB2) For all $p \in P \setminus \{p_0\}$ set $\boxplus'(t_0, p) := \boxplus(t_0, p) + k \cdot \boxplus(t_1, p)$ UB3) Remove p_0 and t_1

Figure 3: Rule B: Sequential place removal (pre)



Precondition	Update
Fix p_0 and t_0, t_1 where $t_0 \neq t_1$ s.t.: B1) ${}^\bullet p_0 = \{t_0\}, p_0^\bullet = \{t_1\}, t_0^\bullet = \{p_0\}$ B2) $\boxplus(t_0, p_0) = k \cdot \boxminus(p_0, t_1)$ for $k \geq 1$ B3) $p_0^\circ = {}^\circ t_0 = {}^\circ t_1 = \emptyset$ B4) $p_0 \notin \mathit{places}(\varphi)$ and $M_0(p_0) = 0$ B5) $p^\circ = \emptyset$ and $p \notin \mathit{places}(\varphi)$ for all $p \in {}^\bullet t_0$	UB1) For all $p \in P \setminus \{p_0\}$ set $\boxminus'(p, t_0) := \boxminus(p, t_0) + k \cdot \boxminus(p, t_1)$ UB2) For all $p \in P \setminus \{p_0\}$ set $\boxplus'(t_0, p) := \boxplus(t_0, p) + k \cdot \boxplus(t_1, p)$ UB3) Remove p_0 and t_1

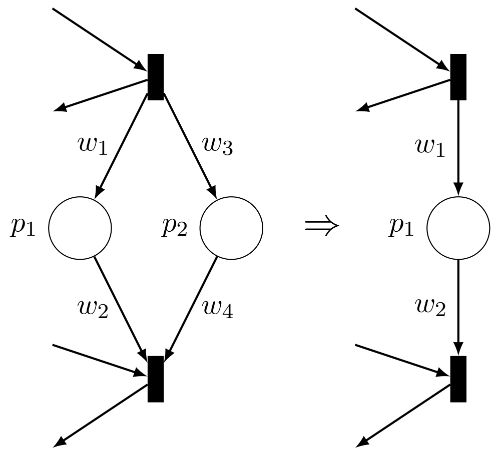
Figure 4: Rule B: Sequential place removal (post)

Rule C: Parallel Places

When two places are symetrically parallel to each other and one may accumulate tokens, Rule C will remove it. See Figure 5. By convention $\min \emptyset = -\infty$ and $\max \emptyset = \infty$. The fraction d describes how fast tokens can be consumed from p_2 compared to p_1, while f describes how slow tokens can be fed to p_2 compared to p_1. If $d \leq f$ then p_2 is always fed faster than it is emptied compared to p_1, which means p_2 can be removed, since it will always be p_1 which is missing tokens and disables their consumers.

Theorem 3 *Rule C shown in Figure 5 are correct for CTL* cardinality properties.*

where
$\frac{w_4}{w_2} \leq \frac{w_3}{w_1}$



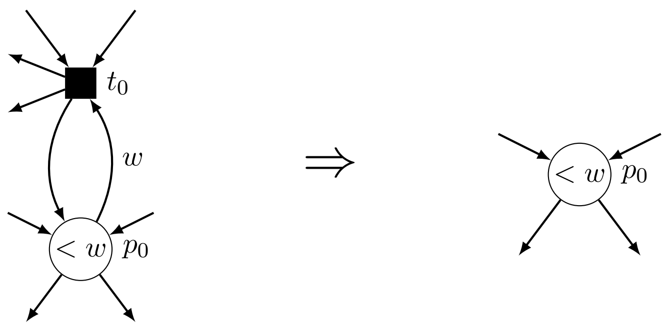
Precondition	Update
Fix places p_1 and p_2 s.t.: C1) $p_2 \notin places(\varphi)$ C2) $p_2^{\circ} = \emptyset$ C3) $p_1^{\bullet} \neq \emptyset$ C4) $p_1^{\bullet} \supseteq p_2^{\bullet}$ C5) ${}^{\bullet}p_1 \subseteq {}^{\bullet}p_2$ C6) $M(p_2) \geq M(p_1) \cdot d$ C7) $d \leq f$ where $d = \max_{t \in p_1^{\bullet}} \frac{\boxminus(p_2, t)}{\boxminus(p_1, t)}$ $f = \min_{t \in {}^{\bullet}p_1} \frac{\boxplus(t, p_2)}{\boxplus(t, p_1)}$	UC1) Remove p_2

Figure 5: Rule C: Parallel places

Rule E: Dead transition removal

If a transition is initially not enabled due to a lack of tokens in p_0 and if p_0 is not able to gain tokens, then the transition is dead and can be removed. See Figure 6.

Theorem 4 *Rule E in Figure 6 is correct for CTL* cardinality properties.*



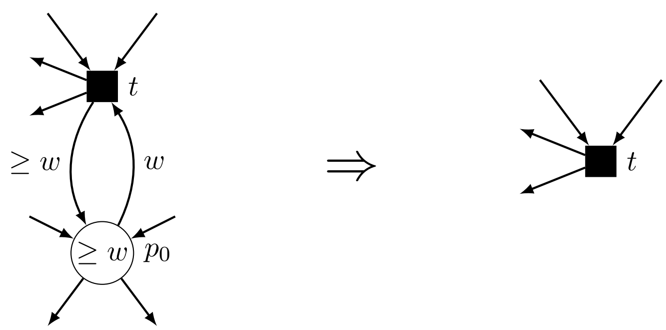
Precondition	Update
Fix place p_0 and transition t_0 s.t.: E1) $M_0(p_0) < \boxminus(p_0, t_0)$ E2) $\boxplus(t, p_0) \leq \boxminus(p_0, t)$ or $M_0(p_0) < \boxminus(p_0, t)$ for all $t \in T$	UE1) If $p_0^\bullet = \{t_0\}, p_0^\circ = \emptyset$, and $p_0 \notin places(\varphi)$ then remove p_0. UE2) Remove t_0

Figure 6: Rule E: Dead transition removal

Rule F: Redundant place removal

Rule F defined in Figure 7 removes places which never inhibits any transitions. This is done by check the minimum number of tokens added to the given place and its initial marking.

Theorem 5 *Rule F in Figure 7 is correct for CTL*.*

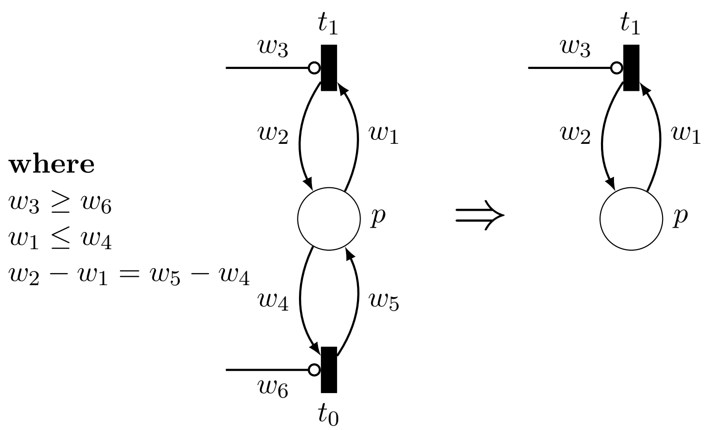


Precondition	Update
Fix place p_0 s.t.: F1) $p_0^\circ = \emptyset$ and $p_0 \notin places(\varphi)$ F2) $\boxplus(t, p_0) \geq \boxminus(p_0, t)$ and $M_0(p_0) \geq \boxminus(p_0, t)$ for all $t \in T$	UF1) Remove p_0

Figure 7: Rule F: Redundant place removal

Rule L: Dominated Transition

Rule L removes transitions that have the same effect as another transition, but with more preconditions. Since both transitions lead to the same state, we can therefore remove the one with the higher preconditions and use the other instead. See the formal description in Figure 8.



Precondition	Update
Fix transition t_1 and t_0 s.t.: L1) $I(t_1) \geq I(t_0)$ L2) $\boxminus(t_1) \leq \boxminus(t_0)$ L3) $E(t_1) = E(t_0)$	UL1) Remove t_0

Figure 8: Rule L: Dominated Transition

Theorem 6 *Rule L in Figure 8 is correct for CTL* cardinality properties.*

Rule M: Effectively dead places and transitions

The Rule M finds and removes effectively dead places and transitions. We define an effectively dead place to be a place that will never gain nor lose tokens. Effectively dead transitions are transitions that are initially disabled (and/or inhibited) by a place that cannot gain (and/or lose) tokens. These places and transitions are found using fixed-point iteration as defined in Algorithm 1.

Algorithm 1: Rule M: Effectively dead places and transitions

Input: A net $N = \langle P, T, \boxminus, \boxplus, I \rangle$, initial marking M_0 and CTL* formula φ

Output: A reduced net N' and its initial marking M'_0

$S_\leq := P$ `/* Places that cannot gain tokens */`
$S_\geq := P$ `/* Places that cannot lose tokens */`
$F := T$ `/* Transitions that cannot fire */`
do
  `/* Find transitions that may fire and update sets accordingly */`
  foreach $t \in F$ *where* $\forall p \in P.(\boxminus(p,t) \leq M_0(p) \vee p \notin S_\leq) \wedge (I(p,t) > M_0(p) \vee p \notin S_\geq)$ **do**
    $F := F \setminus \{t\}$
    $S_\leq := S_\leq \setminus t^{\boxplus}$
    $S_\geq := S_\geq \setminus {}^{\boxminus}t$
until $S_\leq$, $S_\geq$, *and* F *do not change*
$P' := P \setminus (S_\leq \cap S_\geq \setminus places(\varphi))$
$T' := T \setminus F$
return $N' = \langle P', T', \boxminus, \boxplus, I \rangle$ *and* M_0

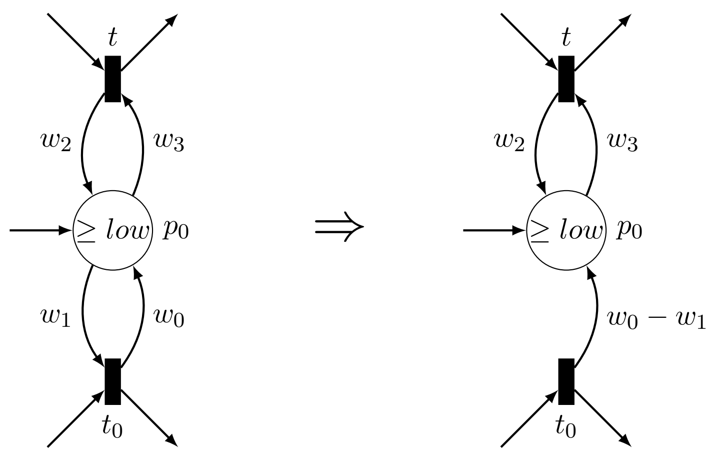
Theorem 7 *Rule M in Algorithm 1 is correct for CTL* cardinality properties.*

Theorem 8 *Rule M supercedes Rule E.*

Rule N: Redundant arc removal

The lower bound number of tokens at a place p_0 is given by the minimum of the initial marking and the number of tokens returned by any consuming transition with a negative effect on p_0. Using the lower bound we can then find transitions, which are never disabled by p_0 and remove the transition's dependency on p_0, since it is unnecessary, as long as we maintain the effect of firing the transition.

where
$w_3 > w_2 \geq low$
$w_0 \geq low \geq w_1$



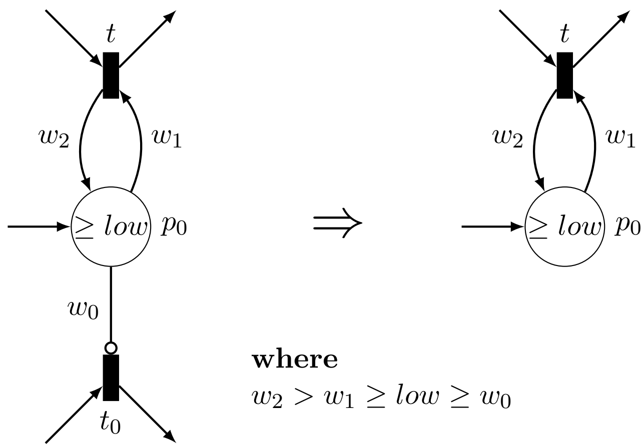
Precondition	Update
Fix place p_0 and transition t_0 s.t.: N1) $t_0 \in p_0^\bullet \setminus p_0^{\boxminus}$ N2) $\boxminus(p_0, t_0) \leq low$ where $low = \min\{M_0(p_0)\} \cup \{\boxplus(p_0, t) \mid t \in p_0^{\boxminus}\}$	UN1) Set $\boxplus(p_0, t_0) := \boxplus(p_0, t_0) - \boxminus(p_0, t_0)$ UN2) Set $\boxminus(p_0, t_0) := 0$

Figure 9: Rule N: Redundant arc removal

Theorem 9 *Rule N in Figure 9 is correct for CTL*.*

Rule O: Inhibited transition

We can find the lower bound of tokens at a place p_0. Any inhibitor arc from p_0 with a weight smaller than the lower bound always inhibits the given transition, which means that the transition can be removed. See Figure 10 for a formal description of Rule O.



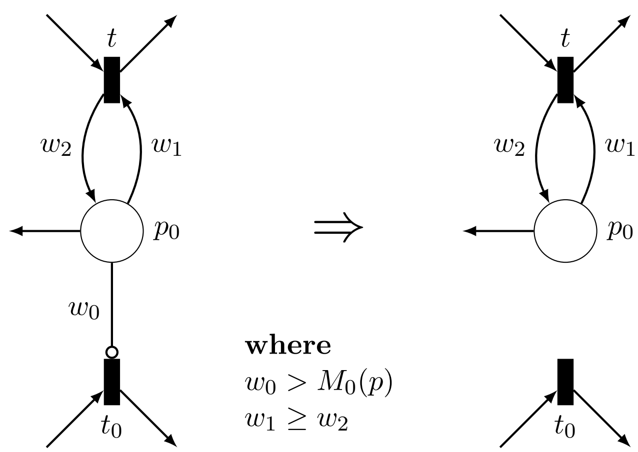
Precondition	Update
Fix place p_0 and transition t_0 s.t.: O1) $t_0 \in p_0^{\circ}$ O2) $I(p_0, t_0) \leq low$ where $low = \min\{M_0(p_0)\} \cup \{\boxplus(p_0, t) \mid t \in p_0^{\boxminus}\}$	UO1) Remove t_0.

Figure 10: Rule O: Inhibited transition

Theorem 10 *Rule O in Figure 10 is correct for CTL* cardinality properties.*

Rule P: Redundant inhibitor arc

Sometimes we can find an upper bound on the number of tokens at a place p_0. This upper bound is given by the initial marking if all transitions have a non-positive effect on p_0. Any inhibitor arc from p_0 with a weight higher than the upper bound of p_0 therefore never inhibits, which means the inhibitor arc can be removed. See Figure 11 for a formal description of Rule P.



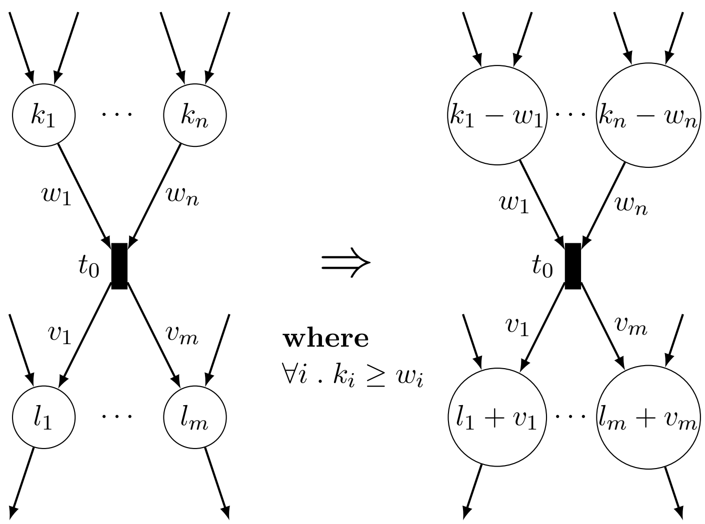
Precondition	Update
Fix place p_0 and transition t_0 s.t.: P1) $t_0 \in p_0^\circ$ P2) $I(p_0, t_0) > M_0(p_0)$ P3) $^{\boxplus}p_0 = \emptyset$	UP1) $I(p_0, t_0) = \infty$.

Figure 11: Rule P: Redundant inhibitor arc

Theorem 11 *Rule P in Figure 11 is correct for CTL*.*

Rule Q: Preemptive transition firing

Rule Q evaluates transitions that are initially enabled and are the only consumer of all places in its pre set. The formal description of Rule Q can be found in Figure 12. Remark that Rule Q can potentially put tokens into places which will prevent other reductions. Furthermore, it can be applied infinitely if $\boxminus(t_0) \leq \boxplus(t_0)$, or if the Petri net contains a loop.



Precondition	Update
Fix transition t_0 s.t.: Q1) $({}^\bullet t)^\bullet = \{t_0\}$ Q2) $\boxminus(t_0) \leq M_0 < I(t_0)$ Q3) $({}^\bullet t_0 \cup t_0^\bullet) \cap places(\varphi) = \emptyset$ Q4) $({}^\bullet t_0)^\circ = (t_0^\bullet)^\circ = \emptyset$	UQ1) $M_0 := M_0 + E(t_0)$.

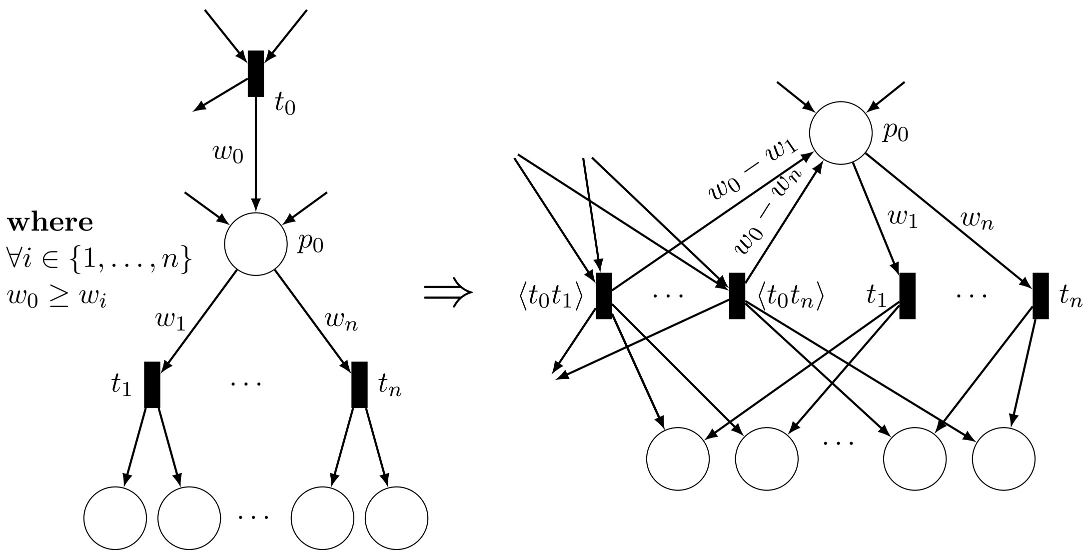
Figure 12: Rule Q: Preemptive transition firing

Theorem 12 *Rule Q in Figure 12 is correct for CTL\X cardinality properties.*

Rule R: Atomic post-agglomerable producer

Rule R is similar to a post agglomeration rule and a formal description of Rule R is in Figure 13. In Rule R we look for a place p_0 with a producer t_0 such that t_0 can always be followed by a firing of any consumer of p_0 without inhibiting other transitions or affecting places in $places(\varphi)$. The producer t_0 is then replaced with new transitions, one for each consumer, and these new transitions combine the effect of firing t_0 and the given consumer. Similarly to an agglomeration rule, Rule R removes interleavings despite potentially increasing the size of the Petri net. However, Rule R is more general, since it only operates on one producer at a time and leaves p_0 untouched, allowing tokens in p_0 in the initial marking, which a post agglomeration does not. Additionally, Rule R does not require the weights of the arcs to and from the agglomerated place to be equal, making R usable in many cases.

Theorem 13 *Rule R in Figure 13 is correct for LTL\X cardinality properties.*



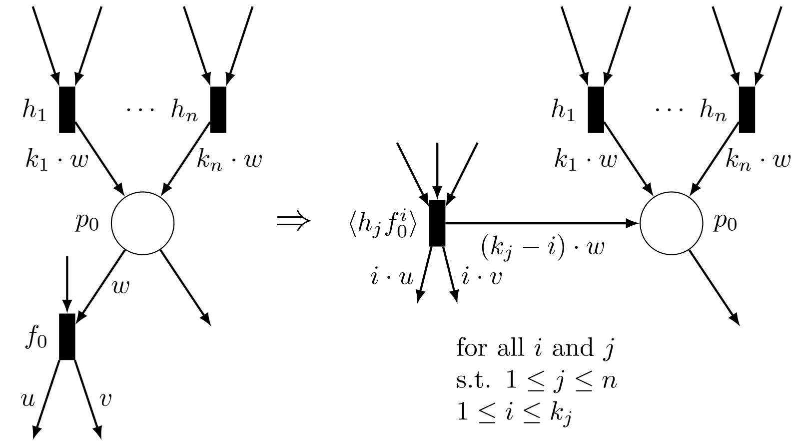
Precondition	Update
Fix place p_0 and transition t_0 s.t.: R1) $t_0 \in {}^\bullet p_0 \wedge p_0^\bullet \neq \emptyset$ R2) ${}^\bullet p_0 \cap p_0^\bullet = \emptyset$ R3) $p_0^\circ = {}^\circ(p_0^\bullet) = ((p_0^\bullet)^\bullet)^\circ = \emptyset$ R4) $(\{p_0\} \cup (p_0^\bullet)^\bullet) \cap places(\varphi) = \emptyset$ R5) ${}^\bullet(p_0^\bullet) = \{p_0\}$ R6) $\boxplus(t_0, p_0) \geq \boxminus(p_0, t)$ for all $t \in p_0^\bullet$	UR1) For each transition $t \in p_0^\bullet$ create a transition $\langle t_0 t \rangle$ with the following arcs: $\boxminus(\langle t_0 t \rangle) = \boxminus(t_0)$ $\boxplus(\langle t_0 t \rangle) = \boxplus(t_0) + \boxplus(t) - \boxminus(t)$ $I(\langle t_0 t \rangle) = I(t_0)$ UR2) Remove t_0

Figure 13: Rule R: Atomic post-agglomerable producer

Rule S: Atomic free agglomeration

A free agglomeration is a pre agglomeration, which does not require that the pre set of the preset of p_0 has a single consumer. In turn, it is only correct for reachability with deadlocks. The atomic free agglomeration is similar to the free agglomeration, but is able to agglomeration one consumer at a time. See Figure 14 for its definition. Rule S also handles cases where the producer h produces k times more tokens than what the consumer f_0 consumes. In this case, a transition $\langle hf_0^i \rangle$ is created for each $i \in [1, k]$. Thus all relevant markings remain reachable.

Theorem 14 *Rule S shown in Figure 14 is correct for deadlock-insensitive reachability properties.*



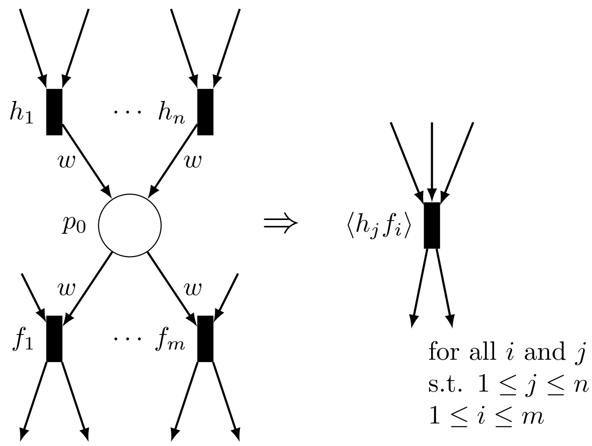
Precondition	Update
Fix place p_0 and transition f_0 s.t.: S1) $\{p_0\} \cap places(\varphi) = \emptyset$ S2) $(f_0 \cup {}^\bullet p_0) \cap transitions(\varphi) = \emptyset$ S3) $M_0(p_0) < \boxminus(p_0, f_0)$ S4) ${}^\bullet p_0 \cap p_0^\bullet = \emptyset$ S5) $f_0 \in p_0^\bullet$ and for all $h \in {}^\bullet p_0$ there exists a $k \in$ s.t.: S6) $h^\bullet = \{p_0\}$ S7) ${}^\bullet h \cap places(\varphi) = \emptyset$ S8) $p_0^\circ = {}^\circ h = ({}^\bullet h)^\circ = \emptyset$ S9) $\boxplus(h, p_0) = k \cdot \boxminus(p_0, f_0)$ S10) $k > 1 \implies (f_0^\bullet)^\circ = \emptyset$ S11) $k > 1 \implies {}^\bullet f_0 = \{p_0\}$	Create transition $\langle h f_0^i \rangle$ for all $i \in [1, k]$, for $k = \boxplus(h, p_0) / \boxminus (p_0, f_0)$, for all $h \in {}^\bullet p_0$. For each such transition: US1) $\boxplus(\langle h f_0^i \rangle, p_0) = \boxplus(h, p_0) - i \cdot \boxminus(p, f_0)$ and for all $p \in P \setminus \{p_0\}$: US2) $\boxminus(p, \langle h f_0^i \rangle) = \boxminus(p, h) \uplus \boxminus(p, f_0)$ US3) $\boxplus(\langle h f_0^i \rangle, p) = i \cdot \boxplus(f_0, p)$ US4) $I(p, \langle h f_0^i \rangle) = I(p, f_0)$ and US5) Remove f_0 US6) If $p_0^\bullet = \emptyset$, remove p_0 and all transitions in ${}^\bullet p_0 \setminus transitions(\varphi)$

Figure 14: Rule S: Atomic free agglomeration

Rule T: Pre agglomearation

Rule T in Figure 15 is a pre agglomeration. In a pre agglomeration $h \in {}^\bullet p_0$ is invisible to the query and once enabled, it stays enabled. Hence, it can be delayed until an $f \in p_0^\bullet$ needs it. Thus Rule T creates a transition $\langle hf \rangle$ for every pair $h \in {}^\bullet p_0$ and $f \in p_0^\bullet$.

Theorem 15 *Rule T described in Figure 15 is correct for LTL\X.*



Precondition	Update
Fix place p_0 s.t.: T1) $(\{p_0\} \cap places(\varphi) = \emptyset$ T2) $(p_0^\bullet \cup {}^\bullet p_0) \cap transitions(\varphi) = \emptyset$ for all $h \in {}^\bullet p_0$ and $f \in p_0^\bullet$: T3) $M_0(p_0) < \boxminus(p_0, f)$ T4) ${}^\bullet p_0 \cap p_0^\bullet = \emptyset$ T5) $({}^\bullet h)^\bullet = \{h\}$ T6) $h^\bullet = \{p_0\}$ T7) ${}^\bullet h \cap places(\varphi) = \emptyset$ T8) $p_0^\circ = {}^\circ h = ({}^\bullet h)^\circ = \emptyset$ T9) $\boxplus(h, p_0) = \boxminus(p_0, f)$	Create transition $\langle hf \rangle$ for all $h \in {}^\bullet p_0$ and $f \in p_0^\bullet$ s.t. for all $p \in P \setminus \{p_0\}$: UT1) $\boxminus(p, \langle hf \rangle) = \boxminus(p, h) + \boxminus(p, f)$ UT2) $\boxplus(\langle hf \rangle, p) = \boxplus(f, p)$ UT3) $I(p, \langle hf \rangle) = I(p, f)$ and UT4) Remove ${}^\bullet p_0$, $p_0^\bullet$ and p_0

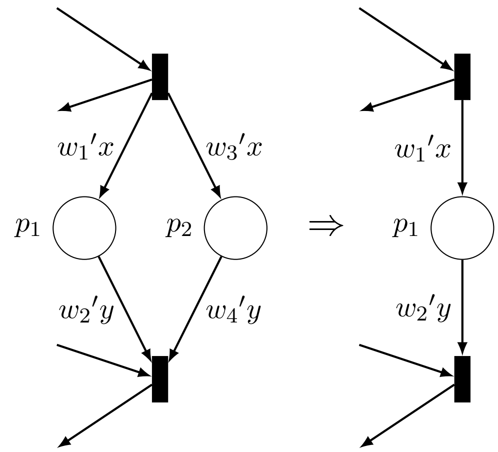
Figure 15: Rule T: Pre agglomeration

Rule C: Parallel Places (CPN)

When two places are symetrically parallel to each other and one may accumulate tokens, Rule C will remove it. See Figure 16. By convention $\min \emptyset = -\infty$ and $\max \emptyset = \infty$. The fraction d describes how fast tokens can be consumed from p_2 compared to p_1, while f describes how slow tokens can be fed to p_2 compared to p_1. If $d \leq f$ then p_2 is always fed faster than it is emptied compared to p_1, which means p_2 can be removed, since it will always be p_1 which is missing tokens and disables their consumers.

Theorem 16 *Rule C shown in Figure 16 are correct for CTL* properties.*

where
$\frac{w_4}{w_2} \leq \frac{w_3}{w_1}$



Precondition	Update
Fix places p_1 and p_2 s.t.: C1) $\mathcal{X}(p_1) = \mathcal{X}(p_2)$ C2) $p_2 \notin places(\varphi)$ C3) $p_2^{\circ} = \emptyset$ C4) $p_1^{\bullet} \neq \emptyset$ C5) For all $t \in T$: $\mathbf{Supp}(\boxminus(p_1,t)) = \mathbf{Supp}(\boxminus(p_2,t)) \wedge$ $\mathbf{Supp}(\boxplus(t,p_1)) = \mathbf{Supp}(\boxplus(t,p_2))$ C6) $\mathbf{Supp}(M_0(p_1)) = \mathbf{Supp}(M_0(p_2)) \wedge$ $M_0(p_1) \cdot d \subseteq M_0(p_2)$ C7) $d \leq f$ where $$d = \max_{t \in p_1^{\bullet}, V \in \boxminus(p_1,t)} \frac{\boxminus(p_2,t)(V)}{\boxminus(p_1,t)(V)}$$ $$f = \min_{t \in {}^{\bullet}p_1, V \in \boxplus(t,p_1)} \frac{\boxplus(t,p_2)(V)}{\boxplus(t,p_1)(V)}$$	UC1) remove p_2

Figure 16: Rule C: Parallel places (CPN)

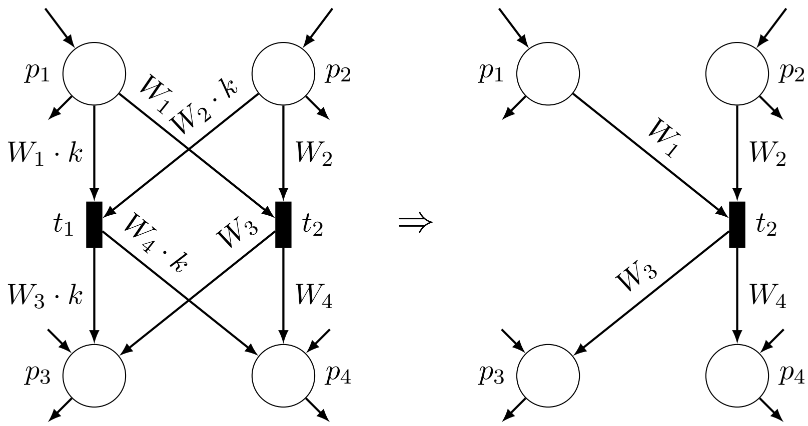
Rule D: Parallel transitions

Rule D handles parallel transitions where the effect of firing one of them is equivalent to firing the other exactly k times. In such a case, we remove the transition with higher arc-weights. The definition of Rule D can be seen in Figure 17. In precondition D2 states that the valid bindings of the guard $G(t_1)$ must be a subset of the valid bindings of $G(t_2)$, i.e. $\vec{B}(t_1) \subseteq \vec{B}(t_2)$. This can be expensive to check depending on the complexity of the guards and the number of variables in the guard. A cheap overapproximation is to check whether $G(t_1) = G(t_2)$ or $G(t_2) = \top$ instead.

Theorem 17 *Rule D described in Figure 17 is correct for LTL$\backslash$X.*

Theorem 18 *Rule D described in Figure 17 is correct for CTL* if $k = 1$.*

where
$k \geq 1$



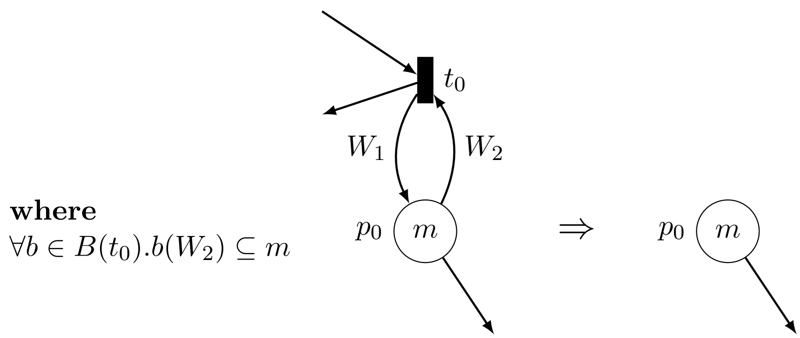
Precondition	Update
Fix transitions t_1 and t_2 and $k \in$ s.t.: D1) $t_1 \notin \mathit{transitions}(\varphi)$ D2) $\vec{B}(t_1) \subseteq \vec{B}(t_2)$ D3) $\varphi \in \mathrm{CTL} \vee X \in \varphi \implies k = 1$ D4) For all $p \in P$: $\boxminus(p, t_1) = \boxminus(p, t_2) \cdot k$ $\boxplus(t_1, p) = \boxplus(t_2, p) \cdot k$ D5) ${}^{\circ}t_2 \cap t_2^{\bullet} = \emptyset$ D6) $\forall p \in P.I(p, t_1) \leq I(p, t_2)$ D7) $\varphi \notin \mathit{Reach} \Rightarrow ({}^{\bullet}t_1 \cup t_1^{\bullet}) \cap (\mathit{places}(\varphi) \cup {}^{\bullet}\mathit{transitions}(\varphi)) = \emptyset$	UD1) remove t_1

Figure 17: Rule D: Parallel transitions

Rule E: Dead transitions

Rule E in Figure 18 removes transitions that are never enabled. If too many bindings exists to check E1, then checking the cardinalities is a valid overapproximation.

Precondition E3 can be ignored φ if all instances of $en(t_0)$ are replaced with $\neg\top$ instead in the update.



Precondition	Update
Fix place p_0 and transition t_0 s.t.:	
E1) $\forall b \in B(t).b(\boxminus(p_0, t_0)) \not\subseteq M_0(p_0)$	UE1) If $p_0 \notin places(\varphi)$ and $p_0^\bullet = \{t_0\}$, remove p_0
E2) $\forall t \in {}^\bullet p_0 : \boxplus(t, p_0) \subseteq \boxminus(p_0, t)$ or $\forall b \in B(t).b(\boxminus(p_0, t)) \not\subseteq M_0(p_0)$	UE2) Remove t_0
E3) $t_0 \notin transitions(\varphi)$	

Figure 18: Rule E: Dead transitions

Theorem 19 *Rule E in Figure 18 is correct for CTL* queries.*