Sorting Networks - Poset Model

Alexandru Lungu
Faculty of Computer Science, UAIC, Iasi

Noiembrie 2020

1 Model

Problema Sorting Networks presupune gasirea unei secvente de comparatori de lungime minima, astfel incat pentru orice intrare σ (σ_i este valoarea de input de pe wire $1 \leq i \leq n$) vom avea un output sortat, deci un $\sigma', \forall 1 \leq i < n, \sigma'_i \leq \sigma'_{i+1}$. n reprezinta dimensiunea instantei.

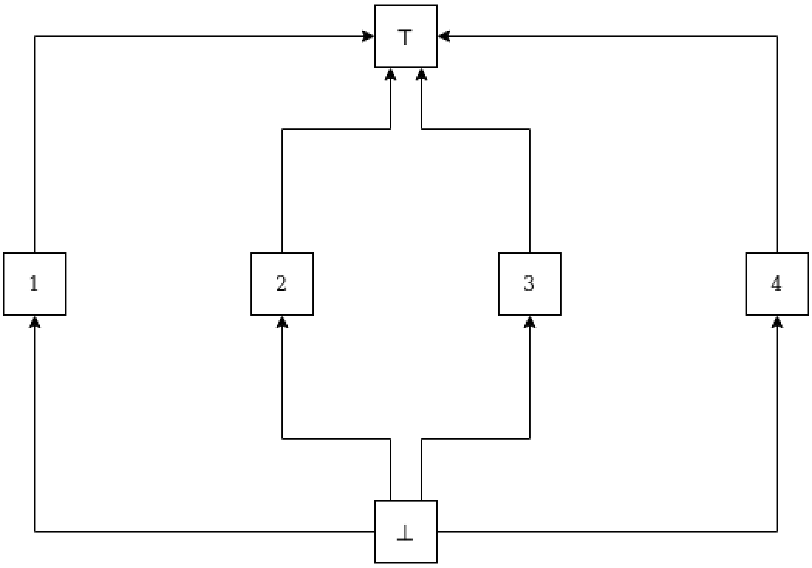
Vom considera un tuplu $(W_n, \leq, C')$, unde:

- Domeniul wire $W_n = \{i | 1 \leq i \leq n\} \cup \{\top, \bot\}$
- Relatia partiala peste wire-uri $x, y \in W_n$
  - $\forall x, y \in W_n \setminus \{\top, \bot\}, (x, y) \in C, x \leq y$
  - $\forall x \in W_n, y = \top, x \leq y$
  - $\forall y \in W_n, x = \bot, x \leq y$
- O multime de relatii $C' \in C_n$, unde C_n este multimea multimilor de relatii peste W_n, $C_n = \{A | A \subseteq \{(x, y) | x, y \in W_n \setminus \{\top, \bot\} \wedge x < y\}$

Pentru simplitate, vom folosi notatia $\bar{W_n} = W_n \setminus \{\top, \bot\}$

2 Motivatie

Vom considera o instanta pentru problema Sorting Networks cu $n = 4$. Modelul unui Network fara comparatori pentru instanta propusa este $(W_4, \leq, \emptyset)$ (in figura de mai jos este reprezentat vizual acest tuplu). Vom presupune ca $L = \{L_{n,k}, \forall n, k \in N\}$



Lema 1 $(W_4, \leq, C_0), C_0 = \emptyset$ *este o latice completa.*

Pentru a determina daca un model este sugerat de un Sorting Network, trebuie sa investigam tipul relatiei de ordine. Daca $\leq$ este o relatie de ordine totala, atunci modelul nostru reprezinta un Sorting Network. Altfel, avem nevoie sa definim "mai bine" $\leq$ pana ajungem la o relatie totala. Prin a defini mai bine, intelegem adaugarea de noi comparatori, ce va duce la o relatie de ordine $\leq$ mai precisa (in figura de mai jos este descrisa grafic o relatie de ordine totala).

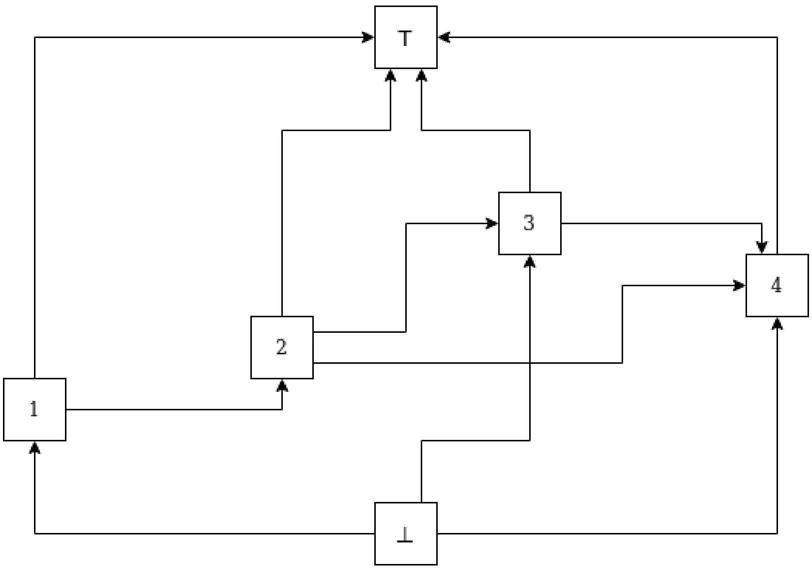


Figura reprezinta modelul $(W_4, \leq, (1,2), (2,3), (2,4), (3,4)\})$

3 Constructie

Pentru a reprezenta o retea de sortare prin intermediul modelului propus, va fi nevoie de o definitie recusiva dupa k pentru $L_{n,k} = (W_n, \leq, C_k)$

- Pasul de baza: $L_{n,0} = (W_n, \leq, C_0), C_0 = \emptyset$
- Presupunem ca la pasul k se va alege un comparator (i, j) si ca $L_{n,k-1} = (W_n, \leq, C_{k-1})$. Atunci $L_{n,k} = [W_n, \leq, \phi_n(L_{n,k-1}, i, j)]$

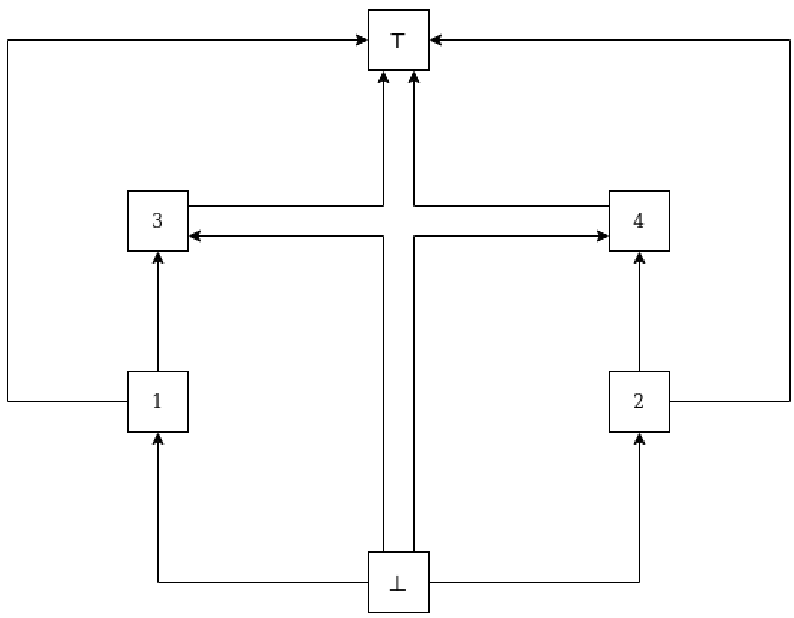
Functia de transfer $\phi_n : L \times (\bar{W_n})^2 \to C$ este o metoda de a genera o multime de relatie potrivita pentru un nou model ce va reprezenta reteaua veche la care se adauga un nou comparator. Vom simplifica notatia pentru functia de transfer si urmatoarele functii prin a omite dependinta fata de dimensiunea instantei. Deci vom folosi in continuare notatia ϕ.

3.1 Definitie pentru functia de transfer triviala

Atunci cand un comparator este luat in considerare, noul model trebuie sa invete o noua relatie astfel: $\phi(L = (W_n, \leq, C'), i, j) = C' \cup learn(L, i, j) \setminus forget(L, i, j)$

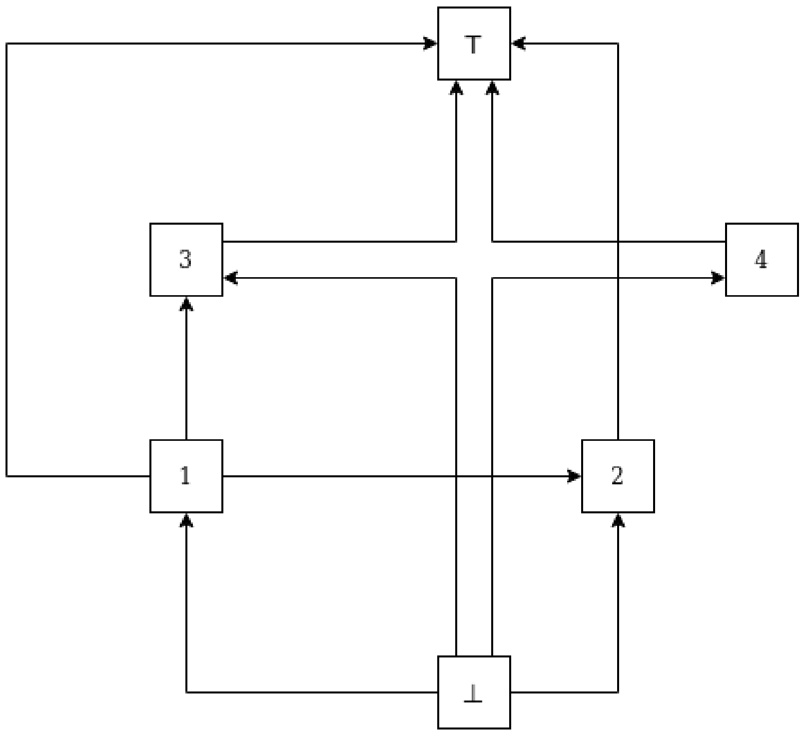
Prin urmare vom avea o functie $learn : L \times (\bar{W})^2 \to 2^{\bar{W} \times \bar{W}}$, ce va fi definita trivial $learn((W_n, \leq, C'), i, j) = \{(i, j)\}$ daca $(i, j) \notin C'$, $learn((W_n, \leq, C'), i, j) = \emptyset$ altfel.

De asemenea, anumite relatii vor trebui sa fie uitate, deoarece in urma aplicarii comparatorului, este posibil ca unele relatii sa nu mai fie corecte. In fapt, vom presupune ca avem un comparator de la i la j $i, j \in \{k | 1 \leq k \leq n\}, i < j$. Vom stii conform $learn$ ca $(i, j) \in C_k$, deci orice $(x, i) \in C_{k-1}$ ar putea fi incorect daca ar fi transferat in C_k. In asemenea masura, orice $(j, x) \in C_{k-1}$ ar putea fi incorect. Din acest motiv vom defini $forget : L \times (\bar{W})^2 \to 2^{\bar{W} \times \bar{W}}$ in felul urmator: $forget((W_n, \leq, C'), i, j) = \{(x, i) \in C'\} \cup \{(j, x) \in C'\}$.



In figura de mai sus este o reprezentare vizuala pentru un model $L_{4,k-1} = (W_4, \leq, C_{k-1}), C_{k-1} = \{(1, 3), (2, 4)\}$ Vom presupune ca vrem sa adaugam comparatorul $(1, 2)$, deci vrem sa calculam

$\phi(L_{4,k-1}, 1, 2)$. Vom obtine $learn(L_{4,k-1}, 1, 2) = \{(1, 2)\}$ si $forget(L_{4,k-1}, 1, 2) = \emptyset \cup \{(2, 4)\}$. In final $\phi(L_{4,k-1}, 1, 2) = \{(1, 2), (1, 3), (3, 4)\}$, deci $L_{4,k} = (W_4, \leq, \{(1, 2), (1, 3), (3, 4)\}$. Acest $L_{4,k}$ este expus grafic mai jos.



Propozitia 1 *Conform definitiei recursive pentru $L_{n,k}$ utilizand functia de transfer triviala, $L_{n,k}$ este o relatie de ordine partiala.*

3.2 Incompletitudinea functiei de transfer triviala

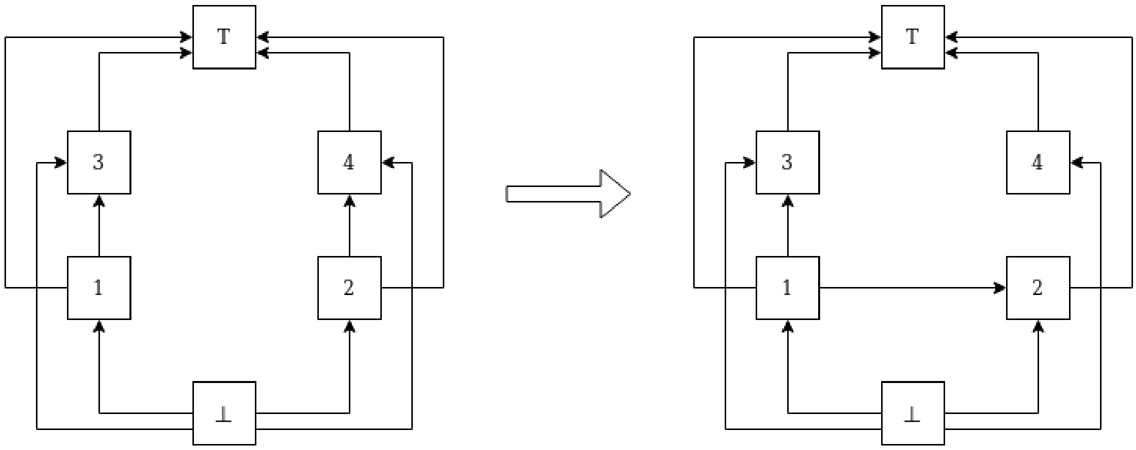
Vom considera exemplul $L_{3,k-1} = (W_3, \leq, \{(1, 3), (1, 2)\})$. Este evident faptul ca, pentru a obtine o relatie de ordine partiala, ne trebuie o relatie intre 2 si 3. Intuitiv ar fi sa adaugam comparatorul $(2, 3)$. Insa $L_{3,k} = (W_3, \leq, \phi(L_{3,k-1}, 2, 3))$ deci $L_{3,k} = (W_3, \leq, \{(1, 3), (2, 3)\}$. Exista totusi o secventa ce sa sa duca la un model sugestiv pentru Sorting Network.

Lema 2 *Nu exista un model $L_{n,k}$ recusiv definit ce sa reprezinte un Sorting Network pentru instante mai mari ca 2, atat timp cat ϕ este o functie de transfer triviala.*

3.3 Definitie pentru functia de transfer cu reinvatare

O problema a functiei de transfer triviala este faptul ca $forget$ include relatii ce ar putea sa fie valide in noul model. Cu alte cuvinte, $forget$ in functia triviala este un upper bound pentru multimea de relatii ce trebuie eliminate.

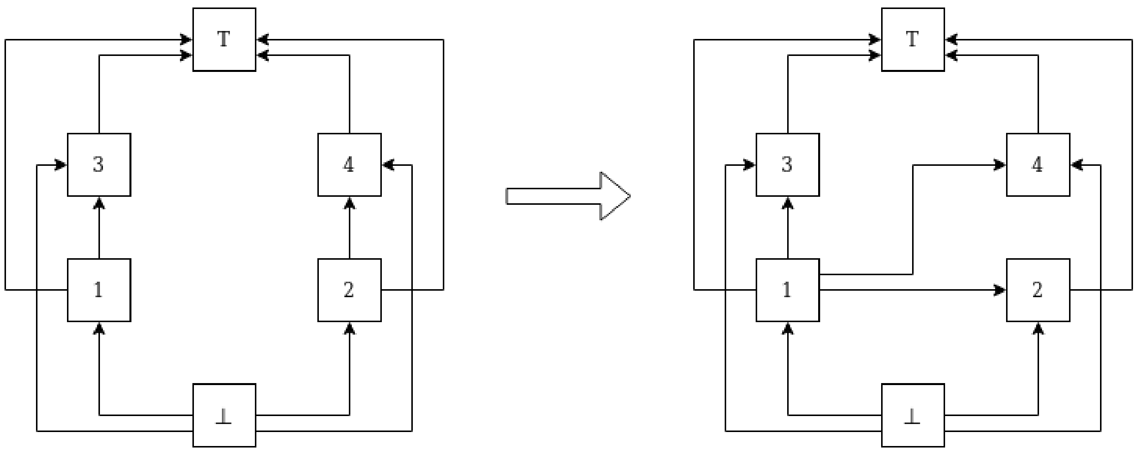
Sa consideram $L_{4,k-1} = (W_4, \leq, \{(1, 3), (2, 4)\})$. Un model $L_{4,k}$ bazat pe adaugarea comparatorului $(1, 2)$ va duce la $L_{4,k} = (W_4, \leq, \{(1, 3), (1, 2)\})$ conform definitiei lui ϕ trivial. Aceasta tranzitie este descrisa in figura de mai jos.



Este evident ca modelul $L_{4,k}$ nu mai ia in considerare $(2,4)$. Aceasta informatie este nesigura incat maximul de pe wire-urile 1 si 2 poate sa nu satisfaca relatia $(2,4)$. In schimb, putem deduce relatia $(1,4)$, incat daca relatia $(2,4)$ era valida iar valoarea din 1 este mai mica ca valoarea din 2 este evident ca valoarea din 1 e mai mica decat valoarea din 4. Prin urmare, redefinim $\phi_{relearn}(L = (W_n, \leq, C'), i, j) = C' \cup learn(L,i,j) \setminus forget(L,i,j) \cup relearn(L,i,j)$.

$$relearn((W_n, \leq, C'), i, j) = \{(x,j) | (x,i) \in C'\} \cup \{(i,x) | (j,x) \in C'\}.$$

Presupunem ca avem o definitie recursiva pentru $L_{n,k}$ folosind functia de transfer cu reinvatare. Pentru exemplul anterior, unde $L_{4,k-1} = (W_4, \leq, \{(1,3),(2,4)\})$ vom deduce ca $L_{4,k} = (W_4, \leq, \phi_{relearn}(L_{n,k}, i, j)$, deci $L_{4,k} = (W_4, \leq, \{(1,3),(2,4)\} \cup \{(1,2)\} \setminus \{(2,4)\} \cup \{(1,4)\}) = (W_4, \leq, \{(1,3),(1,4),(1,2)\})$. Aceasta tranzitie este reprezentata mai jos.



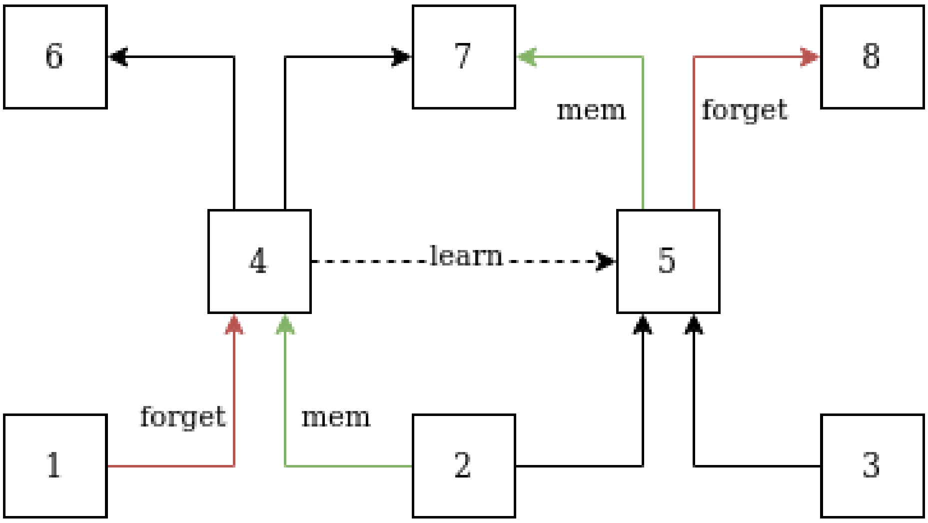
Propozitia 2 *Conform definitiei recursive pentru $L_{n,k}$ utilizand functia de transfer cu reinvatare, $L_{n,k}$ este o relatie de ordine partiala.*

3.4 Definitie pentru functia de transfer cu memoizare

O problema pe care inca o intalnim in functia de transfer cu reinvatare este faptul ca nu retinem anumite relatii ce inca pot fi adevarate. Pentru exemplul din seciunea trecuta $L_{4,k} = (W_4, \leq$

, $\{(1,3),(1,4),(1,2)\}$), putem considera ca un comparator $(2,4)$ nu ar trebui sa determine uitarea unei relatii. Conform definitiilor anterioare pentru functiile de transfer, in urma adaugarii comparatorului $(2,4)$ ar trebui sa uitam relatia $(1,2)$ conform $forget$. Daca luam in considerare ca relatiile $(1,4)$ si $(1,2)$ sunt valide, atunci nu este posibil ca valoarea de pe wire-ul 2 sa ajunga sa fie mai mica decat valoarea de pe wire-ul 1 (deoarece minimul dintre valorile de pe 2 si 4 este mai mare ca valoarea de pe wire-ul 1).

Vom considera urmatoarea subdiagrama pentru modelul nostru.



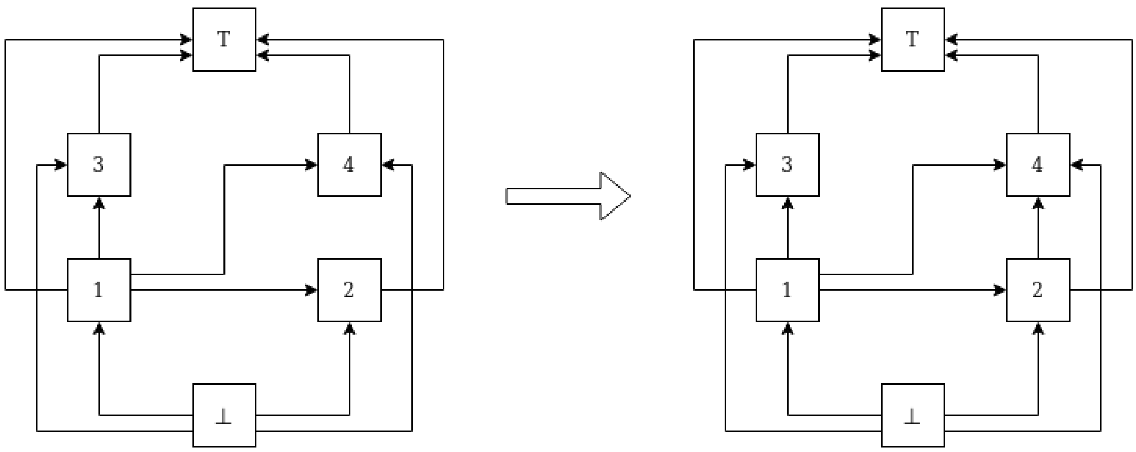
Se observa in figura ca dupa ce invatam relatia $(4,5)$, putem uita relatiile $(1,4),(2,4),(5,7),(5,8)$ cu exceptia celor ce pot fi memoizate, adica $(2,4)$ si $(5,7)$.

Prin urmare vom defini o multime de relatii ce trebuie memoizate:

$$mem((W_n, \leq, C'), i, j) = \{(x,i) \in C' | (x,j) \in C'\} \cup \{(j,x) \in C' | (i,x) \in C'\}$$

Iar functia de transfer de memoizare devine:

$$\phi_{mem}(L_{n,k} = (W_n, \leq, C'), i, j) = C' \cup learn(L,i,j) \setminus forget(L,i,j) \cup relearn(L,i,j) \cup mem(L,i,j)$$

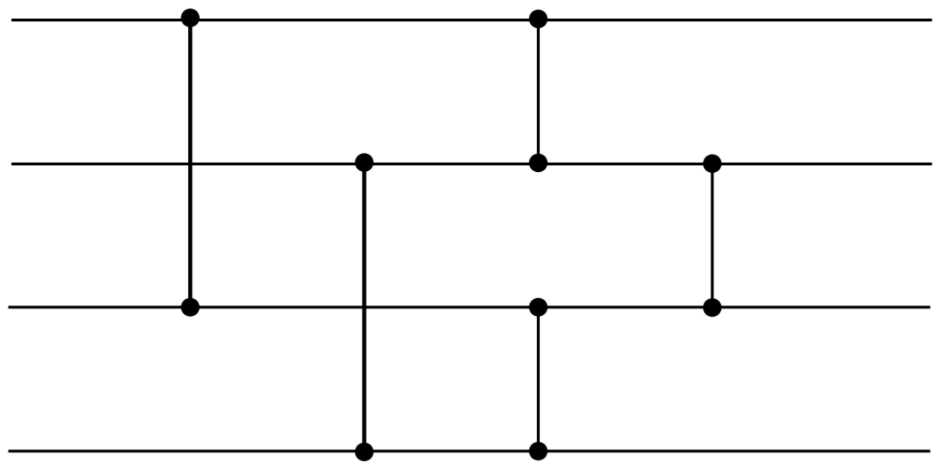


Propozitia 3 *Conform definitiei recursive pentru $L_{n,k}$ utilizand functia de transfer cu memoizare, $L_{n,k}$ este o relatie de ordine partiala.*

Propozitia 4 *Exista un model $L_{n,k}$ recusiv definit ce sa reprezinte un Sorting Network pentru orice instanta, atat timp cat ϕ e ste o functie de transfer cu memoizare.*

4 Neechivalenta

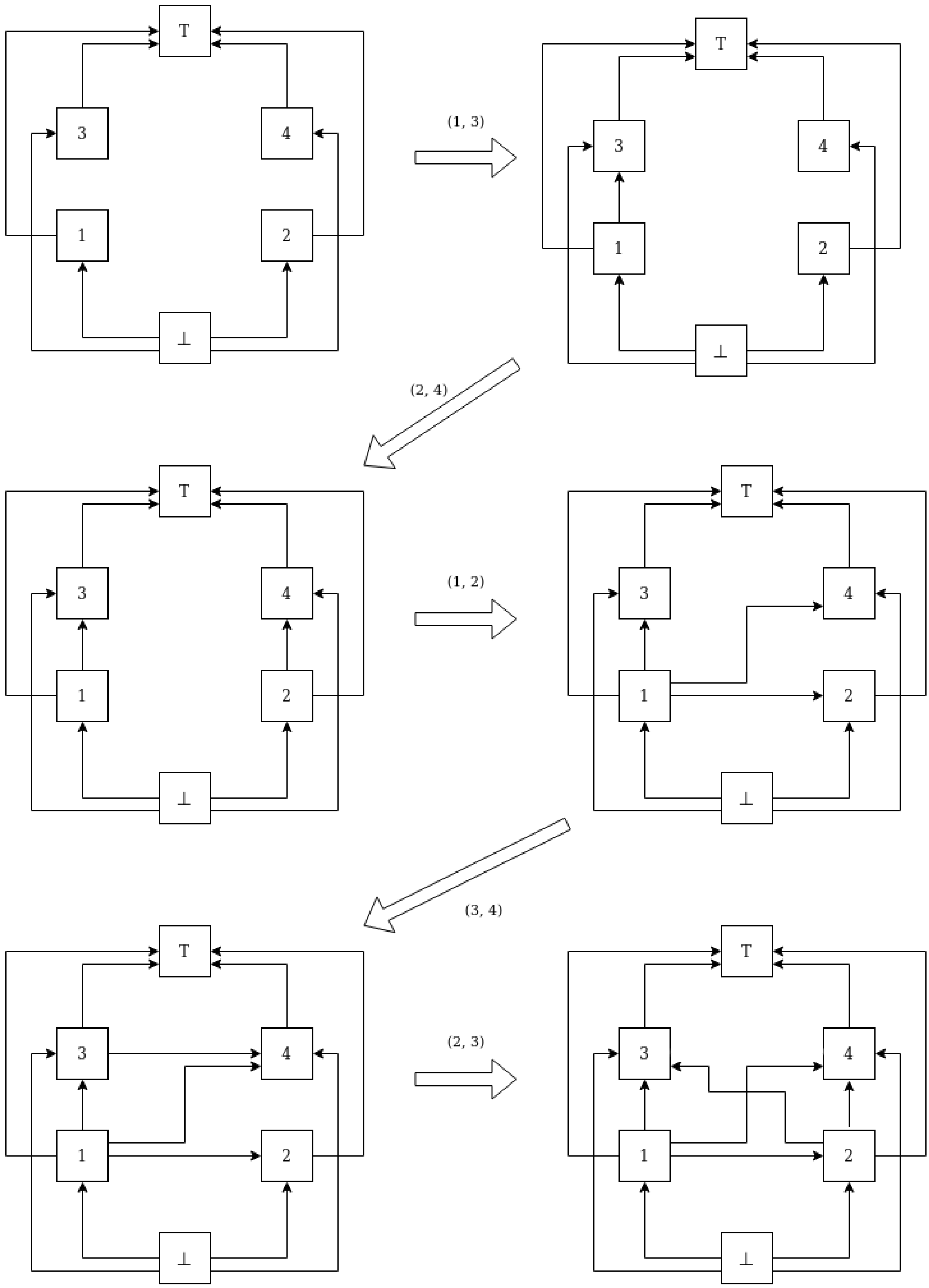
Vom considera in aceasta sectiune o evolutie a modelului conform unei solutii a instantei $n = 4$ din Sorting Network, de este ilustrata mai jos.



Vom observa in figura de mai jos, ca in urma efectuarii unor tranzitii bazate pe functia de transfer cu memoizare, nu am obtinut o relatie de ordine totala, ceea ce inseamna ca nu exista o echivalenta itre model si retea. Insa cu siguranta stim ca daca modelul este o ordonare completa, atunci cu siguranta reteaua este de sortare prin urmare avem urmatoarea lema.

Propozitia 5 *Daca exista o relatie completa $L_{n,k}$ conform definitiei recusive bazate pe functia de transfer cu memoizare, atunci reteaua modelata este o retea de sortare. Reciproca nu este adevarata.*

Lema 3 *Reciproca propzitiei 5 nu este adevarata.*


(1, 3)
(2, 4)
(1, 2)
(3, 4)
(2, 3)

Se observa in model ca relatie finala nu este completa, desi reteaua pe care o modeleaza este de sortare. Acest fapt se datoreaza lemei 3. Comparatorul $(3,4)$ ar determina o ordine totala, deci putem confirma pe baza propzitiei 5 ca reteaua aferenta este de sortare.

Lema 4 *Daca avem $L_{n,k} = (W_n, \leq, \phi(L_{n,k-1}, i, j))$, unde $L_{n,k-1} = (W_n, \leq, C')$, atunci $|\phi(L_{n,k-1}, i, j)| = |C'| + 1$ daca folosim functia de transfer cu reinvatare ori cu memoizare. Asta inseamna, ca la orice pas recursiv, invatam un singur comparator nou.*

Conform lemei 4, pentru a obtine un model complet, atunci avem nevoie de $n * (n-1)/2$ pasi recursivi. Prin urmare, modelul nostru este momentan trivial.

5 Urmarirea valorilor extreme

O optimizare, ce poate determina modelul sa invete mai multe relatii la un pas recursiv, este reprezentata de urmarirea valorilor extreme (maximul si minimul). Aceasta tehnica se bazeaza pe faptul ca valoarea maxima nu va mai fi vreodata pe un wire i ce a facut parte dintr-un comparator $(i,j), i < j$. In aceeasi idee, valoarea minima nu va mai fi vreodata pe un wire j ce a facut parte dintr-un comparator $(i,j), i < j$.

Prin urmare, vom extinde modelul nostru astfel:

$$L_{n,k} = (W_n, \leq, C', Min, Max, low, high)$$

- Multimea de wire-uri candidate ce reprezinta valoarea minima $Min \subseteq \bar{W}$
- Multimea de wire-uri candidate ce reprezinta valoarea maxima $Max \subseteq \bar{W}$
- low este cel mai mic wire pentru care contorizam Min si Max
- $high$ este cel mai mare wire pentru care contorizam Min si Max

Mai jos avem o deinitie recursiva pentru noul model:

$$L_{n,0} = (W_n, \leq, \bar{W}, \bar{W}, 1, n)$$

$$L_{n,k} = (W_n, \leq, \phi(L_{n,k-1}, i_k, j_k), \alpha(L_{n,k-1}, i, j), \beta(L_{n,k-1}, i, j), \gamma(L_{n,k-1}, i, j), \delta(L_{n,k-1}, i, j))$$

- $extract_{min}(A, low, high) = \{(x,i) | x \in A \wedge low < i \leq high\}$ daca $A = \{low\}$ sau $extract_{min}(A) = \emptyset$ altfel
- $extract_{max}(A, low, high) = \{(i,x) | x \in A \wedge low \leq i < high\}$ daca $A = \{high\}$ sau $extract_{max}(A) = \emptyset$ altfel
- $\alpha((W_n, \leq, C', Min, Max, low, high), i, j) = Min \setminus \{j\}$ daca $extract_{min}(Min, low, high) = \emptyset$ sau $\alpha((W_n, \leq, C', Min, Max, low, high), i, j) = \{i | low \leq i \leq n\} \setminus \{y | (x,y) \in \phi_{mem}((W_n, \leq, C'), i, j)\}$ altfel
- $\beta((W_n, \leq, C', Min, Max, low, high), i, j) = Max \setminus \{i\}$ daca $extract_{max}(Max, low, high) = \emptyset$ sau $\beta((W_n, \leq, C', Min, Max, low, high), i, j) = \{i | 1 \leq i \leq high\} \setminus \{x | (x,y) \in \phi_{mem}((W_n, \leq, C'), i, j)\}$ altfel
- $\gamma((W_n, \leq, C', Min, Max, low, high), i, j) = low$ daca $extract_{min}(Min, low, high) = \emptyset$ sau $\gamma((W_n, \leq, C', Min, Max, low, high), i, j) = low + 1$ altfel

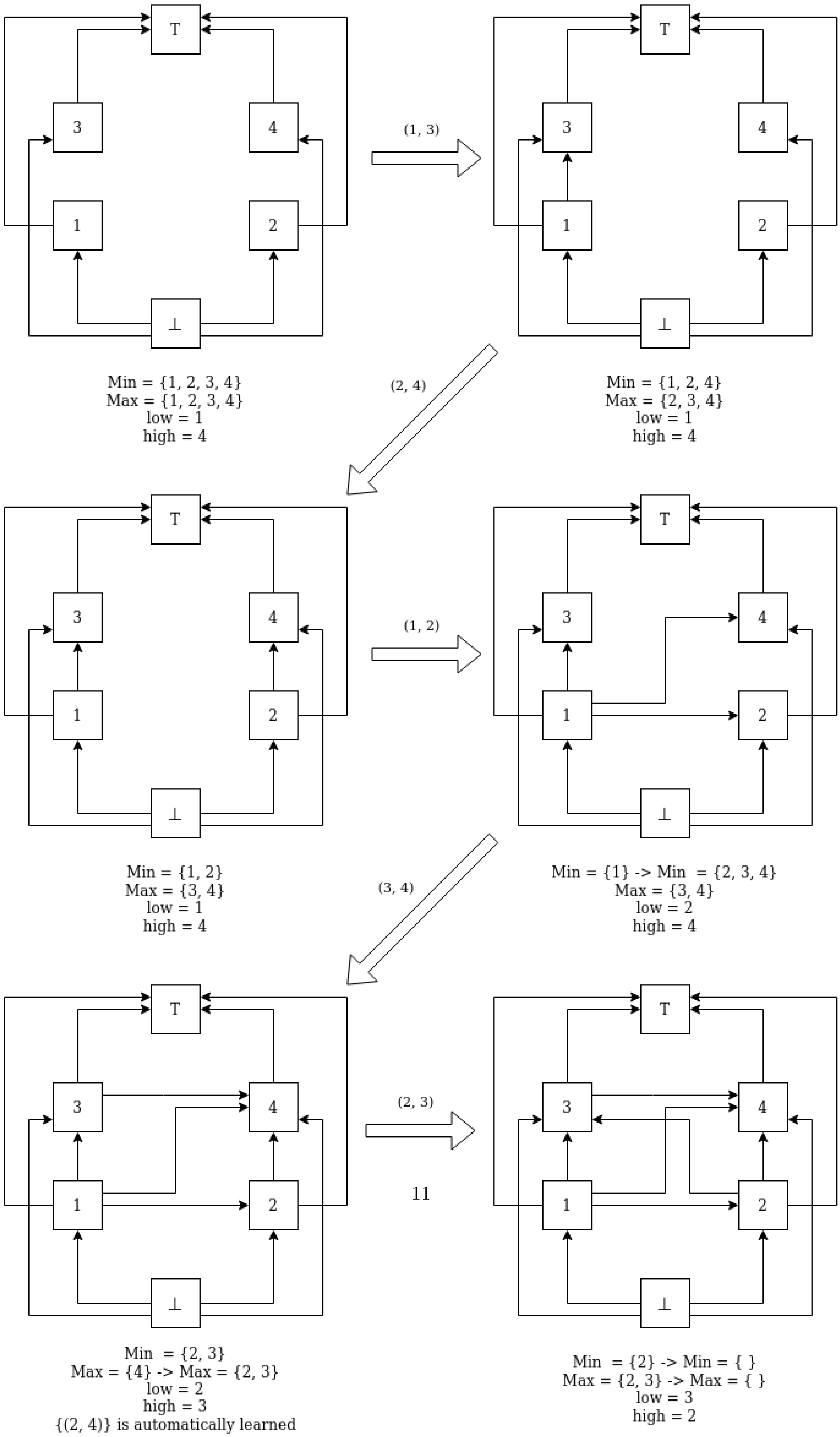
- $\delta((W_n, \leq, C', Min, Max, low, high), i, j) = high$ daca $extract_{max}(Max, low, high) = \emptyset$ sau $\delta((W_n, \leq, C', Min, Max, low, high), i, j) = high - 1$ altfel

Acest principiu se poate extinde si la valorile postminim si premaxim si asa mai departe. Cu alte cuvinte, daca determinam wire-ul ce contine valoarea minima, atunci putem aplica aceeasi procedura pentru restul wire-urilor pentru a determina cine are cea mai mica valoare. Asemanator, vom putea proceda si pentru valorile maxime. Vom redefini functia de transfer astfel:

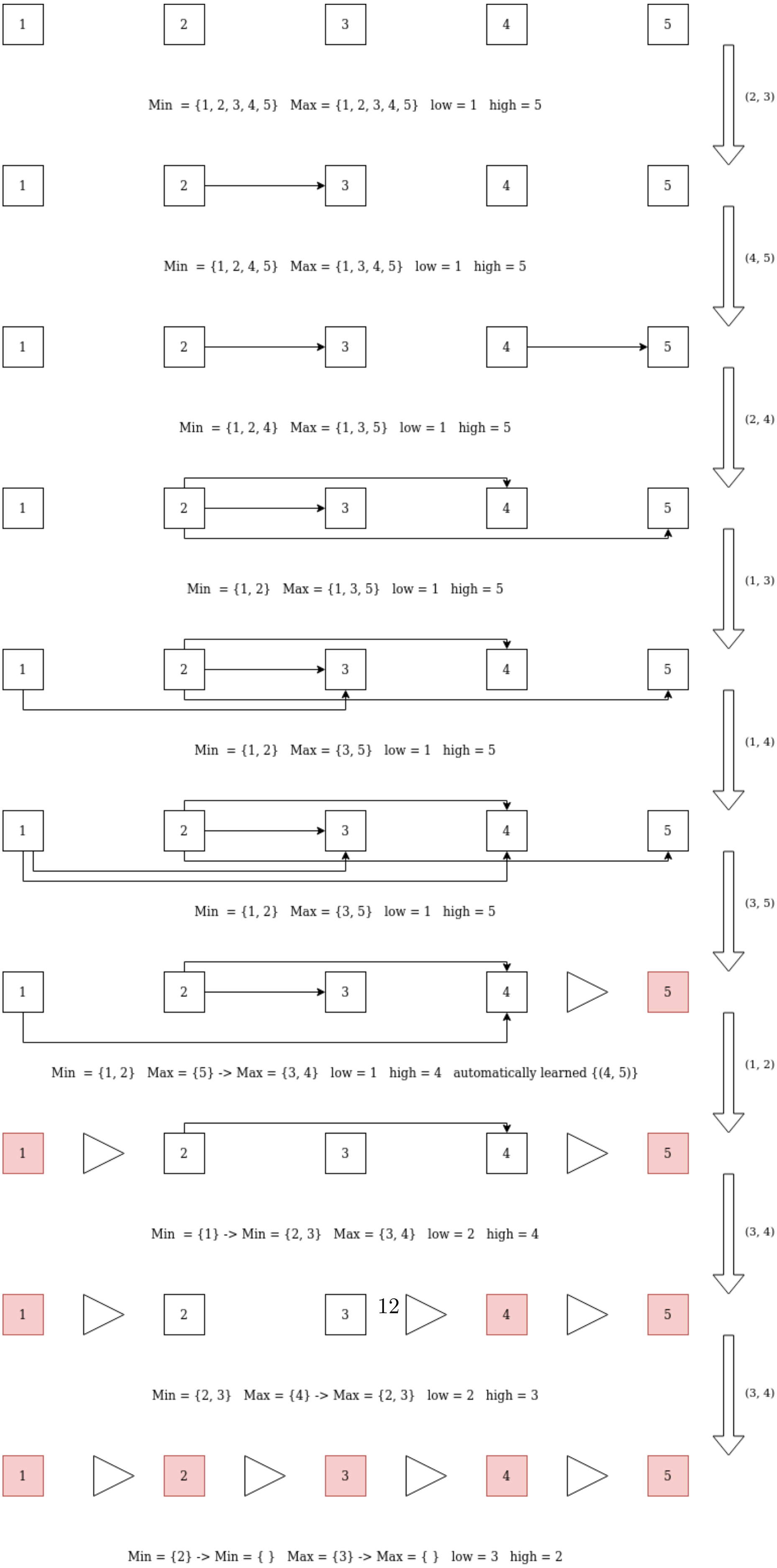
$$\phi_{extreme}(L_{n,k-1} = (W_n, \leq, C', Min, Max, low, high), i, j) = \phi_{mem}((W_n, \leq, C', i, j) \cup extract_{min}(\alpha(L_{n,k-1}, i, j), \gamma(L_{n,k-1}, i, j), \delta(L_{n,k-1}, i, j)) \cup extract_{max}(\beta(L_{n,k-1}, i, j), \gamma(L_{n,k-1}, i, j), \delta(L_{n,k-1}, i, j))$$

5.1 Evolutia modelului pentru instanta n = 4

Pentru instanta n = 4, modelul cu urmarirea valorilor extreme reuseste sa reprezinte o ordine totala atunci cand reteaua aferenta este de sortare. Acest fapt se datoreaza faptului ca avem nevoie de $4 * 3/2 = 6$ relatii pentru ca ordinea sa fie totala. Cei 5 comparatori determina cate o relatie, iar optimizarea urmaririi valorilor extreme ne determina inca o relatie, anume $(2, 4)$ la penultimul pas.


(1, 3)
Min = {1, 2, 3, 4}
Max = {1, 2, 3, 4}
low = 1
high = 4
Min = {1, 2, 4}
Max = {2, 3, 4}
low = 1
high = 4
(2, 4)
(1, 2)
Min = {1, 2}
Max = {3, 4}
low = 1
high = 4
Min = {1} -> Min = {2, 3, 4}
Max = {3, 4}
low = 2
high = 4
(3, 4)
(2, 3)
Min = {2, 3}
Max = {4} -> Max = {2, 3}
low = 2
high = 3
{(2, 4)} is automatically learned
Min = {2} -> Min = { }
Max = {2, 3} -> Max = { }
low = 3
high = 2

5.2 Evolutia modelului pentru instanta n = 5



In diagrama anterioara, am facut abstractie de $\top$ si $\perp$. De asemenea, am facut abstractie de unele relatii atunci cand stim categoric ca un anumit wire are o valoare mai mare ca wire-urile mai mici ca el (ori can un wire are o valoare mai mica ca toate celelalte wire-uri mai mari). Acest lucru este indicat de wire-urile marcater cu rosu. Reteaua de sortare pe care am modelat-o are 9 comparatori (optim pentru instanta $n = 5$). In mod trivial, ar fi fost nevoie de $4 * 5/2 = 10$ relatii pentru a confirma relatia totala. Cu toate acestea, cei 9 comparatori au fost modelati in relatii, iar a 10-a relatie a fost dedusa din optimizarea cu urmariea extremelor.

References