7 Wahlalgorithmen

7.1 Fragestellung

7.2 Ringbasierte Verfahren
- Unidirektionaler Ring
- Bidirektionaler Ring

7.3 Allgemeine Topologien
- Bäume
- Allgemeine Netze
  - Pfadverfahren
  - Kantenfärbung
  - Echo/Wahl-Algorithmen
  - Adoptionsverfahren
  - Warteverfahren







7.2 Ringbasierte Verfahren

Unidirektionaler Ring: Chang-Roberts-Algorithmus

Gewinner ist der Knoten, der unter allen die größte Identifikation besitzt.

Algorithmus für i-ten Prozessor

```
main()
{  typedef enum type {START, ELECT};
   struct message {
      int source, target;
      type messagetype;
      int supposed_leader;
   };
   int supposed_leader = -1;
   int I_am_leader = FALSE;
   message msg;
```







```
   while (1) {
      receive(&msg);
      switch (msg.messagetype)
         case START:
            supposed_leader = i;
            send(i, next(i), ELECT, i);
            break;
         case ELECT:
            if (supposed_leader < msg.supposed_leader) {
               supposed_leader = msg.supposed_leader;
               send(i, next(i), ELECT,
                     msg.supposed_leader);
            } else if (supposed_leader == i) {
               I_am_leader = TRUE;
            }
      }
   }
}
```







Bidirektionaler Ring: modifizierter Chang-Roberts-Algorithmus)

```
main()
{ typedef enum type {START, ELECT};
  struct message {
    int   source, target;
    type  messagetype;
    int   supposed_leader;
    int   d;      /* +1 oder -1 als Richtungsangabe */
  };
  int d, supposed_leader = -1, I_am_leader = FALSE;
  message msg;
```

## Wahlalgorithmen: Unidirektionaler Ring

```
  while (1) {
    receive(&msg);
    switch (msg.messagetype)
      case START:
        supposed_leader = i;
        d = mit_gleicher_Wahrsch_eine_der_Richtungen();
        send(i, next(i, d) , ELECT, i, d);
        break;
      case ELECT:
        if (supposed_leader < msg.supposed_leader) {
          supposed_leader = msg.supposed_leader;
          send(i,next(i,msg.d),ELECT,msg.supposed_leader,msg.d);
        } else if (supposed_leader == i)
          I_am_leader = TRUE;
    }
  }
}
```


Universität Erlangen-Nürnberg, IMMD IV, F. Hofmann





## Wahlalgorithmen: Wahl auf Bäumen

### 7.3 Allgemeine Verfahren

### 7.3.1 Wahl auf Bäumen

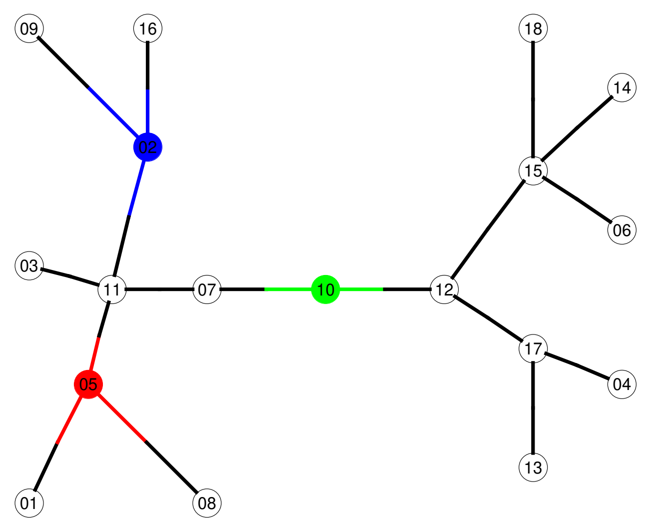
1. Ein Starter sendet *Explosionsnachrichten* in alle Richtungen.
2. Ein (innerer) Knoten, der erstmalig eine Explosionsnachricht erhält, sendet eine solche in alle anderen Richtungen.
3. Ein Blattknoten, der eine Explosionsnachricht erhält, sendet eine *Kontraktionsnachricht* mit seiner eigenen Identität zurück.
4. Ein innerer Knoten, der von allen bis auf einen Nachbarn eine Kontraktionsnachricht erhalten hat, sendet eine Kontraktionsnachricht mit dem Maximum aus den empfangenen Identitäten und seiner eigenen Identität über die restliche Kante.
5. Erhält ein Knoten eine Kontraktionsnachricht nach dem Versenden seiner eigenen Kontraktionsnachricht, so sendet er das Maximum der damit erhaltenen Information und der ihm bereits bekannten Information in Form von *Informationsnachrichten* in alle anderen Richtungen.
6. Erhält ein Knoten eine Informationsnachricht, so propagiert er diese über alle anderen Kanten.


Universität Erlangen-Nürnberg, IMMD IV, F. Hofmann





## Wahlalgorithmen: Wahl auf Bäumen



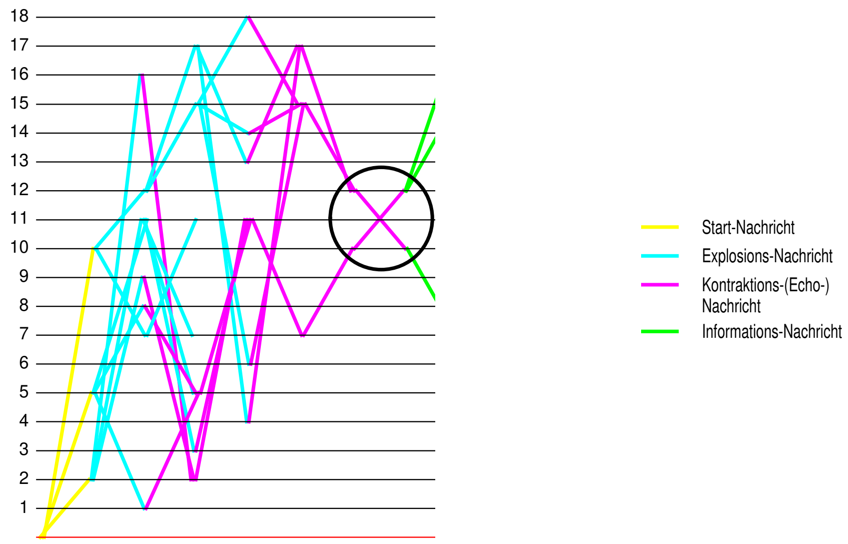


Universität Erlangen-Nürnberg, IMMD IV, F. Hofmann





## Wahlalgorithmen: Wahl auf Bäumen

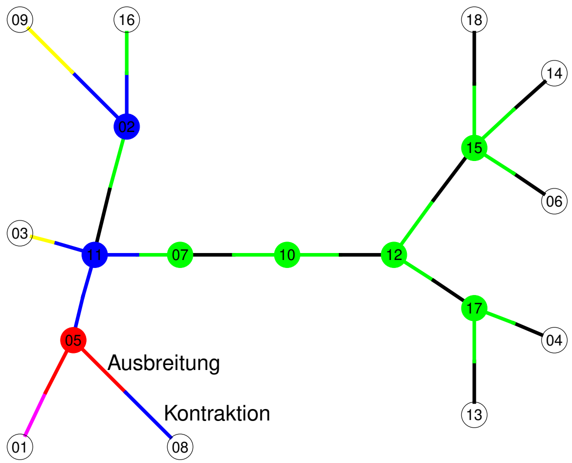




Universität Erlangen-Nürnberg, IMMD IV, F. Hofmann

## Wahlalgorithmen: Wahl auf Bäumen

Farbe charakterisiert Kenntnisstand
(Farbwahl: Knotennummer modulo 6)



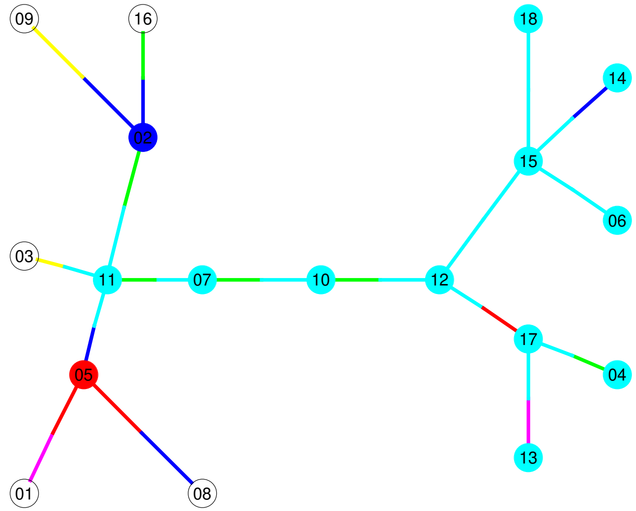


Universität Erlangen-Nürnberg, IMMD IV, F. Hofmann





## Wahlalgorithmen: Wahl auf Bäumen





Universität Erlangen-Nürnberg, IMMD IV, F. Hofmann





## Wahlalgorithmen: Das Pfadverfahren

### 7.3.2 Das Pfadverfahren (depth first)

**Das Token enthält den Pfad zurück zum Initiator und die Angabe des Initiators**

**Die Knoten kennen ihre Nachbarn, halten ihren jeweiligen Kenntnisstand bezüglich des Gewinners fest und notieren, welchen Nachbarn sie ihren neuesten Kenntnisstand schon bekannt gemacht haben.**

**Nachrichten: m = (Empfänger, Pfad, Initiator)**

```
1. Ein Knoten erhält ein Startsignal:
   Falls bislang nicht besucht:

     if (Gewinner < eigene_Nummer) {
       Gewinner = eigene_Nummer;
       Sende(Nachbar, Pfad(eigene_Nummer), eigene_Nummer);
     }
```


Universität Erlangen-Nürnberg, IMMD IV, F. Hofmann





## Wahlalgorithmen: Das Pfadverfahren

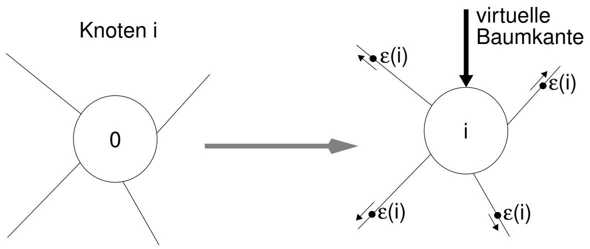
```
2. Ein Knoten erhält die Nachricht m:

    Vorgänger = m.pfad.first();
    if (m.Initiator >= Gewinner) {
      Gewinner = m.Initiator;
      if (noch_nicht_an_alle_Nachbarn_mit_Ausnahme_des_
          _Vorgängers_jetzigen_Gewinner_bekannt_gemacht) {
          Sende(einer_dieser_Nachbarn,
                eigene_Nummer + m.Pfad,
                m.Initiator);
      } else if (m.Pfad != LEERER_PFAD) {
        Sende(Vorgänger, m.Pfad.tail(), m.Initiator);
      } else {
        Dieser_Knoten_ist_Gewinner_der_Wahl;
      }
    }
```


Universität Erlangen-Nürnberg, IMMD IV, F. Hofmann

## 7.3.3 Das Echo-Wahlverfahren

**Gewinner ist der Initiator, der die größte Identifikation besitzt**

**(1) Ein freier Knoten i wird spontan zum Initiator:**





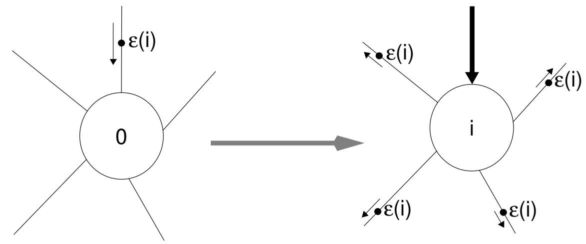
Universität Erlangen-Nürnberg, IMMD IV, F. Hofmann



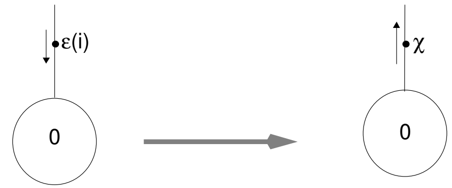


**(2) Ein freier Knoten wird von einem Explorer ε(i) erreicht:**

**a) Der Knoten besitzt einen Grad größer als 1:**




**b) Der Knoten besitzt den Grad 1:**





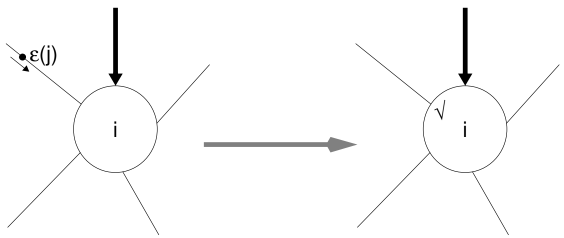
Universität Erlangen-Nürnberg, IMMD IV, F. Hofmann





**(3) Ein Explorer ε(j) trifft auf einen bereits mit i markierten Knoten (mit Grad größer 1):**

**a) j = i, d. h. zwei Explorer identischer Markierung begegnen sich**




Falls die letzte Nicht-Baumkante als erledigt fest-
gestellt wird, wird ein Echo über die Baumkante verschickt.

**b) j < i, d. h. der eintreffende Explorer ist kleiner als der zuletzt versandte:**

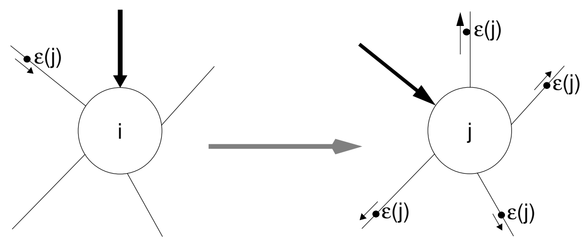
**Es erfolgt keine Zustandsänderung!**


Universität Erlangen-Nürnberg, IMMD IV, F. Hofmann





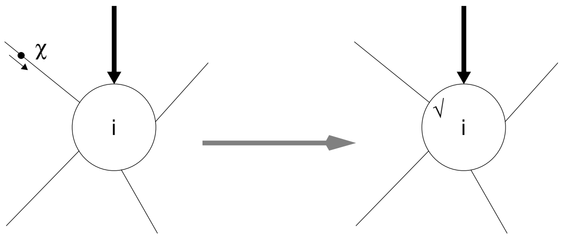
**c) j > i, d. h. Knoten wird von stärkerem Initiator erobert**




Falls ein Initiator erobert wird, entfällt
damit dessen virtuelle Baumkante.


Universität Erlangen-Nürnberg, IMMD IV, F. Hofmann

**(4) Es trifft ein Echo ein:**



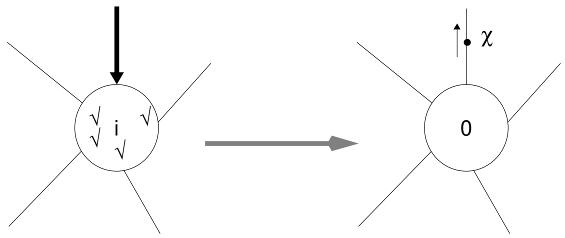

Falls die letzte Nicht-Baumkante als erledigt festgestellt wird, wird ein Echo über die Baumkante verschickt.


Universität Erlangen-Nürnberg, IMMD IV, F. Hofmann





**(5) Zurücksenden eines Echos:**

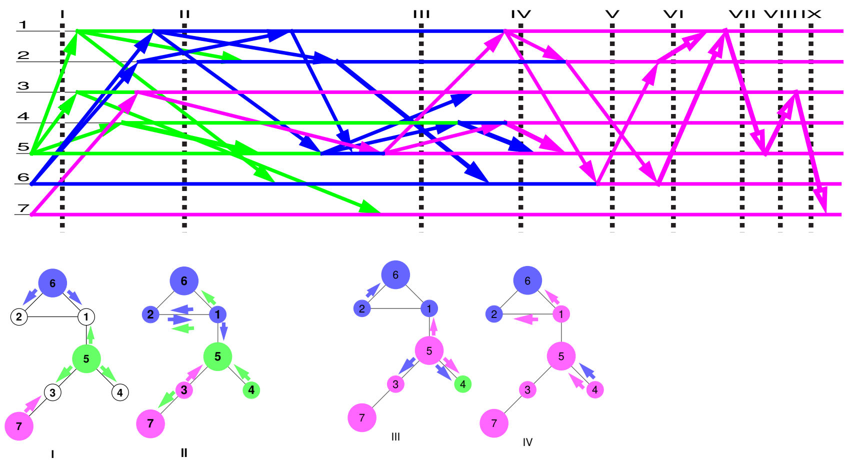



Falls es sich um einen Initiator handelt, d. h. die Baumkante ist virtuell, so ist er der (vorläufige) Gewinner.


Universität Erlangen-Nürnberg, IMMD IV, F. Hofmann









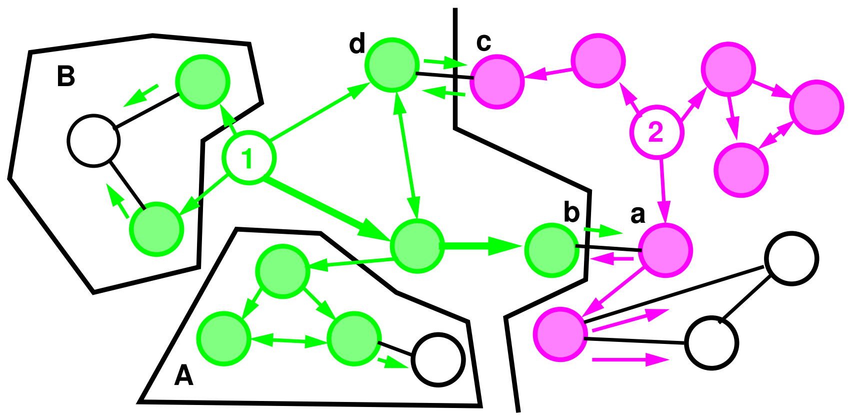
Universität Erlangen-Nürnberg, IMMD IV, F. Hofmann





## 7.3.4 Das Adoptionsverfahren

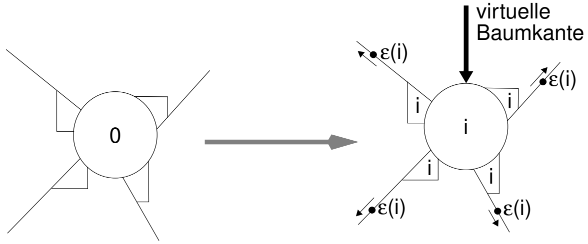
**Die Idee**





Universität Erlangen-Nürnberg, IMMD IV, F. Hofmann

**(1) Ein freier Knoten i wird spontan zum Initiator:**



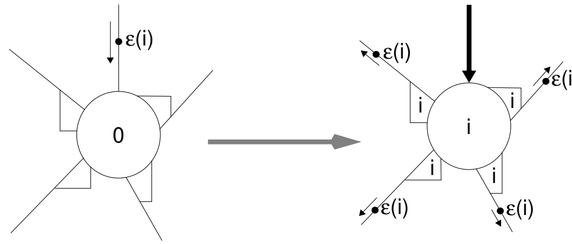




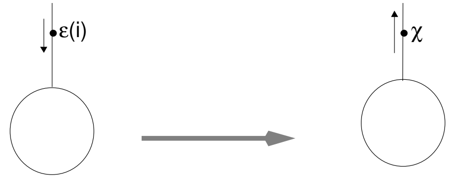



**(2) Ein freier Knoten wird von einem Explorer ε(i) erreicht:**

**a) Der Knoten besitzt einen Grad größer als 1:**




**b) Der Knoten besitzt den Grad 1:**



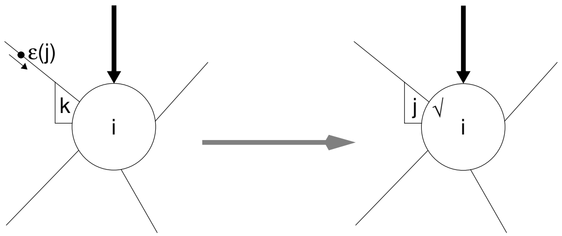







**(3) Ein Explorer ε(j) trifft auf einen bereits mit i markierten Knoten (mit Grad größer 1). Die Eingangskante sei mit k markiert:**

**a) k = j, d. h. zwei Explorer identischer Markierung begegnen sich**




Falls die letzte Nicht-Baumkante als erledigt festgestellt wird, wird ein Echo über die Baumkante verschickt.



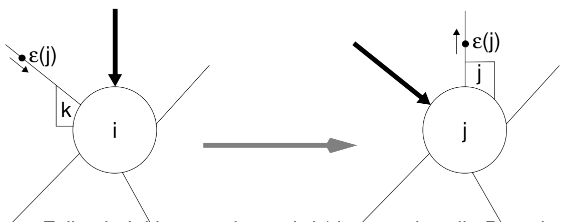




**b) k > j, d. h. der eintreffende Explorer ist kleiner als der in Gegenrichtung zuletzt versandte:**

**Es erfolgt keine Zustandsänderung!**

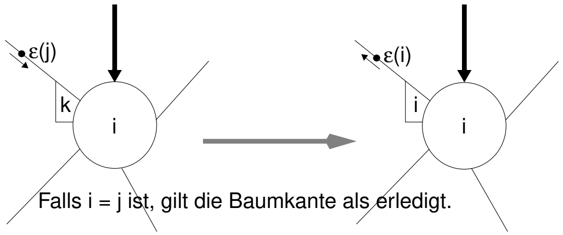
**c) k < j und j > i**




Falls ein Initiator erobert wird (dessen virtuelle Baumkante damit entfällt) wird ebenfalls ein Echo zurückgesandt.

## Wahlalgorithmen: Adoptionsverfahren

**d) k < j und j <= i**




Falls i = j ist, gilt die Baumkante als erledigt.

Handelt es sich dabei um die letzte Nicht-Baumkante wird ein Echo über die Baumkante verschickt.

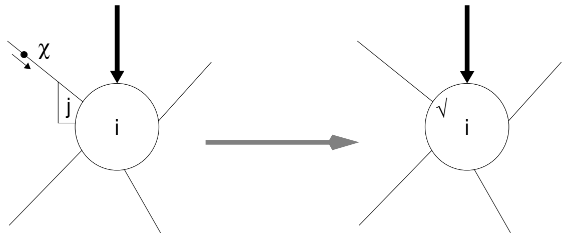

Universität Erlangen-Nürnberg, IMMD IV, F. Hofmann





## Wahlalgorithmen: Adoptionsverfahren

**(4) Es trifft ein Echo ein:**




Falls die letzte Nicht-Baumkante als erledigt festgestellt wird, wird ein Echo über die Baumkante verschickt.

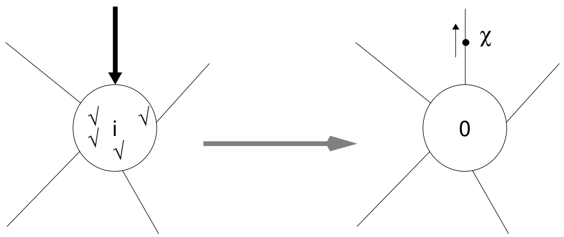

Universität Erlangen-Nürnberg, IMMD IV, F. Hofmann





## Wahlalgorithmen: Adoptionsverfahren

**(5) Zurücksenden eines Echos:**



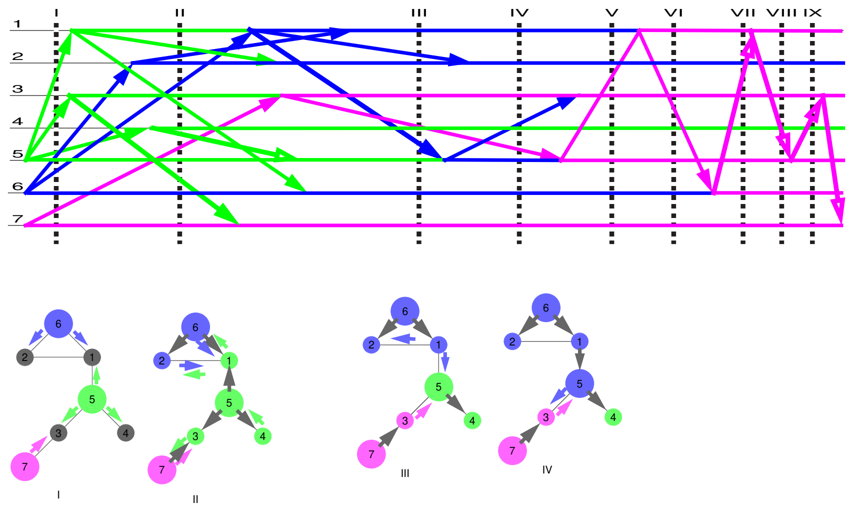

Falls es sich um einen Initiator handelt, d. h. die Baumkante ist virtuell, so ist er der Gewinner.


Universität Erlangen-Nürnberg, IMMD IV, F. Hofmann





## Wahlalgorithmen: Adoptionsverfahren

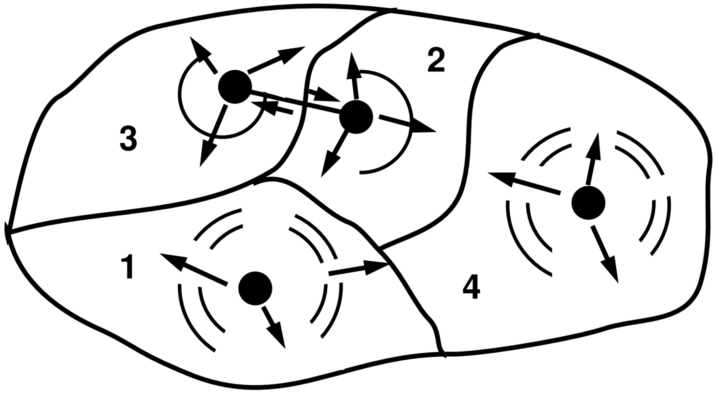




Universität Erlangen-Nürnberg, IMMD IV, F. Hofmann

## 7.3.5 Das Warteverfahren

### Die Idee



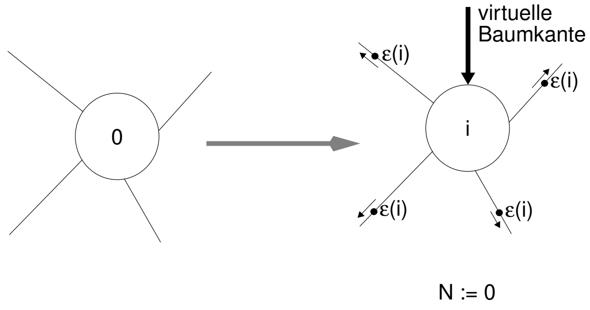


Universität Erlangen-Nürnberg, IMMD IV, F. Hofmann





**(1) Ein freier Knoten i wird spontan zum Initiator:**





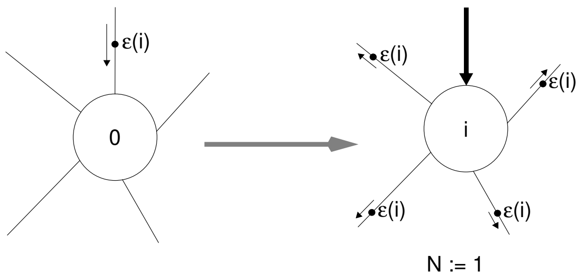
Universität Erlangen-Nürnberg, IMMD IV, F. Hofmann





**(2) Ein freier Knoten wird von einem Explorer ε(i) erreicht:**

**a) Der Knoten besitzt einen Grad größer als 1:**



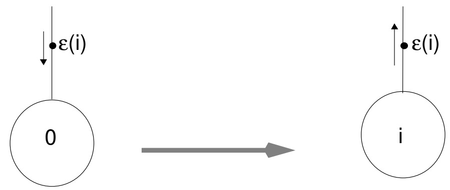


Universität Erlangen-Nürnberg, IMMD IV, F. Hofmann





**b) Der Knoten besitzt den Grad 1:**





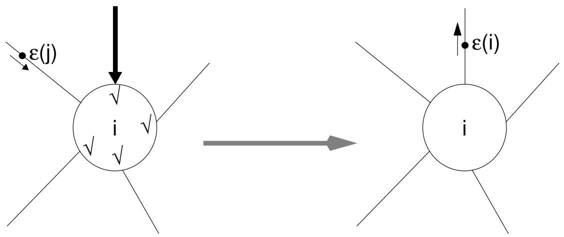
Universität Erlangen-Nürnberg, IMMD IV, F. Hofmann

(3) Ein Explorer ε(j) trifft auf einen bereits mit i markierten Knoten.
In diesem Fall wird zunächst generell N um 1 erhöht.

a) Grad des Knotens = N, es wartet kein Explorer vor dem Knoten, die Baumkante ist eine virtuelle Kante:

Der Knoten hat die Wahl gewonnen.

b) Grad des Knotens = N, es wartet kein Explorer vor dem Knoten, die Baumkante ist echte Kante:



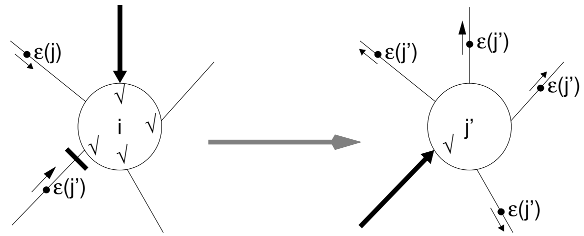







c) Grad des Knotens = N, es wartet bereits ein Explorer ε(j') vor dem Knoten:

```
N = Wartezähler;
Wartezähler = 0;
j' = 0;
```




d) Grad des Knotens ≠ N:

Keine Zustandsänderung







(4) Ein Explorer ε(j) trifft auf einen mit i > j markierten Knoten:

Keine Zustandsänderung. (Explorer läuft auf)

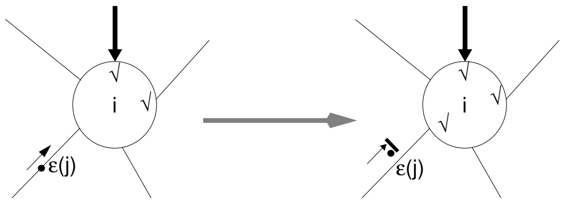
(5) Ein Explorer ε(j) trifft auf einen mit i < j markierten Knoten:

a) Knotengrad = 1: Reaktion wie im Fall 2b.

b) Knotengrad > 1

α) N = 1 und Initiator: Wie 2

β)




```
N++;
if (j > j')  { Wartezähler = 1; j' = j; }
   else if (j == j'){ Wartezähler++; }
```

Gegebenenfalls noch 3c.