

A Network Coding Algorithm for Multi-Layered Video Streaming

Zoltán Király

Department of Computer Science and Communication Networks Laboratory
Eötvös Loránd University, Budapest

Joint work with **Erika Kovács**, EgRes, Eötvös University.

The 2011 International Symposium on Network Coding, Beijing



www.leaderstudio.net

Problem description

- Increasing demand for on-line video downloads, internet TV broadcasting, teleconferencing. . .

www.leaderstudio.net

Problem description

- Increasing demand for on-line video downloads, internet TV broadcasting, teleconferencing. . .
- different available download speeds and/or output resolutions of receivers.

www.leaderstudio.net

Problem description

- Increasing demand for on-line video downloads, internet TV broadcasting, teleconferencing. . .
- different available download speeds and/or output resolutions of receivers.
- Need: service, where different quality requirements are taken into account effectively.

www.leaderstudio.net

Problem description

- Increasing demand for on-line video downloads, internet TV broadcasting, teleconferencing. . .
- different available download speeds and/or output resolutions of receivers.
- Need: service, where different quality requirements are taken into account effectively.
- One practical solution: multi-layered video streaming.

www.leaderstudio.net

- Different realizations,

Multi-resolution codes

- Different realizations,
- here we focus on **multi-resolution codes**.

www.leaderstudio.net

Multi-resolution codes

- Different realizations,
- here we focus on **multi-resolution codes**.
- One base layer, and some refinement layers.

www.leaderstudio.net

Multi-resolution codes

- Different realizations,
- here we focus on **multi-resolution codes**.
- One base layer, and some refinement layers.
- Receivers request cumulative layers,

www.leaderstudio.net

- Different realizations,
- here we focus on **multi-resolution codes**.
- One base layer, and some refinement layers.
- Receivers request cumulative layers,
- they combine them to provide progressive refinement.

- Different realizations,
- here we focus on **multi-resolution codes**.
- One base layer, and some refinement layers.
- Receivers request cumulative layers,
- they combine them to provide progressive refinement.
- The decoding of a higher layer requires the correct reception of

Multi-resolution codes

- Different realizations,
- here we focus on **multi-resolution codes**.
- One base layer, and some refinement layers.
- Receivers request cumulative layers,
- they combine them to provide progressive refinement.
- The decoding of a higher layer requires the correct reception of all lower layers including the base layer.

www.leaderstudio.net

MRC, Base layer = Layer 1



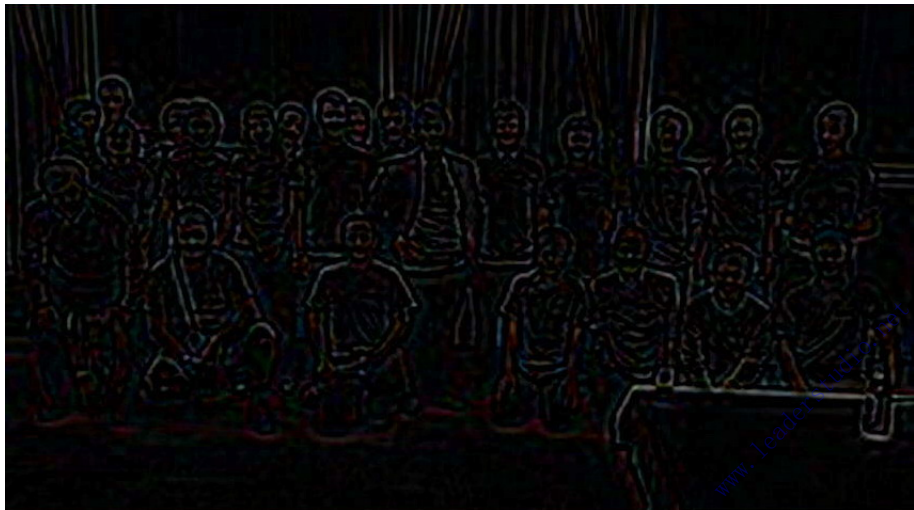
Watching the same on a mobile phone:



www.leaderstudio.net

MRC, Base layer = Layer 1





MRC, Layers 1 + 2



MRC, Layers 1 + 2 + 3



- Goal: use of network coding to improve performance.

- Goal: use of network coding to improve performance.
- NC for multi-resolution codes: many papers about separate coding of layers,

- Goal: use of network coding to improve performance.
- NC for multi-resolution codes: many papers about separate coding of layers,
- first polynomial algorithm for inter-layer NC:

- Goal: use of network coding to improve performance.
- NC for multi-resolution codes: many papers about separate coding of layers,
- first polynomial algorithm for inter-layer NC:
- M. Kim, D. E. Lucani, X. Shi, F. Zhao and M. Médard: "Network coding for multi-resolution multicast", INFOCOM, 2010.

- Goal: use of network coding to improve performance.
- NC for multi-resolution codes: many papers about separate coding of layers,
- first polynomial algorithm for inter-layer NC:
- M. Kim, D. E. Lucani, X. Shi, F. Zhao and M. Médard: "Network coding for multi-resolution multicast", INFOCOM, 2010.
- Nice, simple and distributed heuristic,

- Goal: use of network coding to improve performance.
- NC for multi-resolution codes: many papers about separate coding of layers,
- first polynomial algorithm for inter-layer NC:
- M. Kim, D. E. Lucani, X. Shi, F. Zhao and M. Médard: "Network coding for multi-resolution multicast", INFOCOM, 2010.
- Nice, simple and distributed heuristic,
- only a very weak theorem was proved:

- Goal: use of network coding to improve performance.
- NC for multi-resolution codes: many papers about separate coding of layers,
- first polynomial algorithm for inter-layer NC:
- M. Kim, D. E. Lucani, X. Shi, F. Zhao and M. Médard: "Network coding for multi-resolution multicast", INFOCOM, 2010.
- Nice, simple and distributed heuristic,
- only a very weak theorem was proved:
- using their algorithm, every receiver can decode the base layer.

- $D = (V, A)$: acyclic digraph with source s and unit capacity arcs

- $D = (V, A)$: acyclic digraph with source s and unit capacity arcs
- k : number of layers

Notations

- $D = (V, A)$: acyclic digraph with source s and unit capacity arcs
- k : number of layers
- \mathbb{F}_q^k : vector space over the q element field, unit vectors: $\mathbf{e}_1, \dots, \mathbf{e}_k$

www.leaderstudio.net

- $D = (V, A)$: acyclic digraph with source s and unit capacity arcs
- k : number of layers
- \mathbb{F}_q^k : vector space over the q element field, unit vectors: $\mathbf{e}_1, \dots, \mathbf{e}_k$
- $\mathbf{M} = (M_1, \dots, M_k)$: actual packets to send
($M_i \in \mathbb{F}_q$)

- $D = (V, A)$: acyclic digraph with source s and unit capacity arcs
- k : number of layers
- \mathbb{F}_q^k : vector space over the q element field, unit vectors: $\mathbf{e}_1, \dots, \mathbf{e}_k$
- $\mathbf{M} = (M_1, \dots, M_k)$: actual packets to send ($M_i \in \mathbb{F}_q$)
- $\mathbf{c} : A \rightarrow \mathbb{F}_q^k$ called **network code**, if fulfills the linear combination property:

- $D = (V, A)$: acyclic digraph with source s and unit capacity arcs
- k : number of layers
- \mathbb{F}_q^k : vector space over the q element field, unit vectors: $\mathbf{e}_1, \dots, \mathbf{e}_k$
- $\mathbf{M} = (M_1, \dots, M_k)$: actual packets to send ($M_i \in \mathbb{F}_q$)
- $\mathbf{c} : A \rightarrow \mathbb{F}_q^k$ called **network code**, if fulfills the linear combination property:

for $u \neq s$: $\mathbf{c}(uv) \in \langle \{\mathbf{c}(wu) \mid wu \in A\} \rangle =: \langle \mathbf{c}, u \rangle$

- $D = (V, A)$: acyclic digraph with source s and unit capacity arcs
- k : number of layers
- \mathbb{F}_q^k : vector space over the q element field, unit vectors: $\mathbf{e}_1, \dots, \mathbf{e}_k$
- $\mathbf{M} = (M_1, \dots, M_k)$: actual packets to send ($M_i \in \mathbb{F}_q$)
- $\mathbf{c} : A \rightarrow \mathbb{F}_q^k$ called **network code**, if fulfills the linear combination property:
for $u \neq s$: $\mathbf{c}(uv) \in \langle \{\mathbf{c}(wu) \mid wu \in A\} \rangle =: \langle \mathbf{c}, u \rangle$
- **Note:** on arc a , the message sent is $\mathbf{c}(a) \cdot \mathbf{M}$

- Given network code \mathbf{c} , a node v can decode the message M_i , if $\mathbf{e}_i \in \langle \mathbf{c}, v \rangle$.

Definitions

- Given network code \mathbf{c} , a node v **can decode** the message M_i , if $\mathbf{e}_i \in \langle \mathbf{c}, v \rangle$.
- A layer M_i is **valuable** for a node, if messages M_1, \dots, M_i are all decodable.

Definitions

- Given network code \mathbf{c} , a node v **can decode** the message M_i , if $\mathbf{e}_i \in \langle \mathbf{c}, v \rangle$.
- A layer M_i is **valuable** for a node, if messages M_1, \dots, M_i are all decodable.
- $\lambda(\mathbf{s}, v)$ denotes the maximum number of arc-disjoint paths from s to v .

Definitions

- Given network code \mathbf{c} , a node v **can decode** the message M_i , if $\mathbf{e}_i \in \langle \mathbf{c}, v \rangle$.
- A layer M_i is **valuable** for a node, if messages M_1, \dots, M_i are all decodable.
- $\lambda(s, v)$ denotes the maximum number of arc-disjoint paths from s to v .
- A **request** of a node can be the first i layers.

Definitions

- Given network code \mathbf{c} , a node v **can decode** the message M_i , if $\mathbf{e}_i \in \langle \mathbf{c}, v \rangle$.
- A layer M_i is **valuable** for a node, if messages M_1, \dots, M_i are all decodable.
- $\lambda(s, v)$ denotes the maximum number of arc-disjoint paths from s to v .
- A **request** of a node can be the first i layers.
- T_i denotes the set of nodes with request i .

Definitions

- Given network code \mathbf{c} , a node v **can decode** the message M_i , if $\mathbf{e}_i \in \langle \mathbf{c}, v \rangle$.
- A layer M_i is **valuable** for a node, if messages M_1, \dots, M_i are all decodable.
- $\lambda(s, v)$ denotes the maximum number of arc-disjoint paths from s to v .
- A **request** of a node can be the first i layers.
- T_i denotes the set of nodes with request i .
- A **demand** τ is a sequence of disjoint subsets of $V \setminus \{s\}$ denoted by $\tau = (T_1, T_2, \dots, T_k)$.

Definitions

- Given network code \mathbf{c} , a node v **can decode** the message M_i , if $\mathbf{e}_i \in \langle \mathbf{c}, v \rangle$.
- A layer M_i is **valuable** for a node, if messages M_1, \dots, M_i are all decodable.
- $\lambda(\mathbf{s}, v)$ denotes the maximum number of arc-disjoint paths from \mathbf{s} to v .
- A **request** of a node can be the first i layers.
- T_i denotes the set of nodes with request i .
- A **demand** τ is a sequence of disjoint subsets of $V \setminus \{\mathbf{s}\}$ denoted by $\tau = (T_1, T_2, \dots, T_k)$.
- A demand is **proper**, if $\lambda(\mathbf{s}, t) \geq i$ for all i and all $t \in T_i$.

Definitions

- Given network code \mathbf{c} , a node v **can decode** the message M_i , if $\mathbf{e}_i \in \langle \mathbf{c}, v \rangle$.
- A layer M_i is **valuable** for a node, if messages M_1, \dots, M_i are all decodable.
- $\lambda(s, v)$ denotes the maximum number of arc-disjoint paths from s to v .
- A **request** of a node can be the first i layers.
- T_i denotes the set of nodes with request i .
- A **demand** τ is a sequence of disjoint subsets of $V \setminus \{s\}$ denoted by $\tau = (T_1, T_2, \dots, T_k)$.
- A demand is **proper**, if $\lambda(s, t) \geq i$ for all i and all $t \in T_i$.
- A network code is **feasible** for demand τ if for all i and for every receiver node $t \in T_i$

Definitions

- Given network code \mathbf{c} , a node v **can decode** the message M_i , if $\mathbf{e}_i \in \langle \mathbf{c}, v \rangle$.
- A layer M_i is **valuable** for a node, if messages M_1, \dots, M_i are all decodable.
- $\lambda(s, v)$ denotes the maximum number of arc-disjoint paths from s to v .
- A **request** of a node can be the first i layers.
- T_i denotes the set of nodes with request i .
- A **demand** τ is a sequence of disjoint subsets of $V \setminus \{s\}$ denoted by $\tau = (T_1, T_2, \dots, T_k)$.
- A demand is **proper**, if $\lambda(s, t) \geq i$ for all i and all $t \in T_i$.
- A network code is **feasible** for demand τ if for all i and for every receiver node $t \in T_i$
 t can decode M_j for all $j \leq i$.

Theorem

Given D and a demand $\tau = (T_1, \emptyset, T_3)$, it is NP-complete to decide, whether there exists a feasible network code for this demand.

www.leaderstudio.net

Theorem

Given D and a demand $\tau = (T_1, \emptyset, T_3)$, it is NP-complete to decide, whether there exists a feasible network code for this demand.

Theorem

Given D , a demand $\tau = (T_1, T_2)$ and a number K , it is NP-complete to decide whether there exists a network code satisfying at least K requests.

www.leaderstudio.net

- The **height** of a linear code on an arc is the least important layer with nonzero coefficient on this arc.

- The **height** of a linear code on an arc is the least important layer with nonzero coefficient on this arc.
- For example, vector $(1, 0, 1, 0)$ has height 3.

- The **height** of a linear code on an arc is the least important layer with nonzero coefficient on this arc.
- For example, vector $(1, 0, 1, 0)$ has height 3.
- A function $f : A \rightarrow \{1, 2, \dots, k\}$ is called an **arc-limit**.

- The **height** of a linear code on an arc is the least important layer with nonzero coefficient on this arc.
- For example, vector $(1, 0, 1, 0)$ has height 3.
- A function $f : A \rightarrow \{1, 2, \dots, k\}$ is called an **arc-limit**.
- Arc-limit f is **realizable** for a proper demand τ ,

- The **height** of a linear code on an arc is the least important layer with nonzero coefficient on this arc.
- For example, vector $(1, 0, 1, 0)$ has height 3.
- A function $f : A \rightarrow \{1, 2, \dots, k\}$ is called an **arc-limit**.
- Arc-limit f is **realizable** for a proper demand τ , if there is a feasible network code \mathbf{c} with height $= f(a)$ for every arc a .

Theorem

Arc-limit $f : A \rightarrow \{1, 2\}$ is realizable for a proper demand $\tau = (T_1, T_2)$, iff for all arcs $uv \in A$, $u \neq s$

www.leaderstudio.net

Theorem

Arc-limit $f : A \rightarrow \{1, 2\}$ is realizable for a proper demand $\tau = (T_1, T_2)$,
iff for all arcs $uv \in A$, $u \neq s$

- 1 if $f(uv) = 2$, then $\exists wu \in A : f(wu) = 2$,

Theorem

Arc-limit $f : A \rightarrow \{1, 2\}$ is realizable for a proper demand $\tau = (T_1, T_2)$,
iff for all arcs $uv \in A$, $u \neq s$

- 1 if $f(uv) = 2$, then $\exists wu \in A : f(wu) = 2$,
- 2 if $f(uv) = 1$, then either $\exists wu \in A : f(wu) = 1$, or $\lambda(s, u) \geq 2$, and moreover

Theorem

Arc-limit $f : A \rightarrow \{1, 2\}$ is realizable for a proper demand $\tau = (T_1, T_2)$, iff for all arcs $uv \in A$, $u \neq s$

- 1 if $f(uv) = 2$, then $\exists wu \in A : f(wu) = 2$,
- 2 if $f(uv) = 1$, then either $\exists wu \in A : f(wu) = 1$, or $\lambda(s, u) \geq 2$, and moreover
- 3 for any receiver $t \in T_1$ with $\lambda(s, t) = 1$, there is a 1-limited arc entering t , and

Theorem

Arc-limit $f : A \rightarrow \{1, 2\}$ is realizable for a proper demand $\tau = (T_1, T_2)$, iff for all arcs $uv \in A$, $u \neq s$

- 1 if $f(uv) = 2$, then $\exists wu \in A : f(wu) = 2$,
- 2 if $f(uv) = 1$, then either $\exists wu \in A : f(wu) = 1$, or $\lambda(s, u) \geq 2$, and moreover
- 3 for any receiver $t \in T_1$ with $\lambda(s, t) = 1$, there is a 1-limited arc entering t , and
- 4 for any $t \in T_2$ there is a 2-limited arc entering t .

www.leadstudio.net

Theorem

Arc-limit $f : A \rightarrow \{1, 2\}$ is realizable for a proper demand $\tau = (T_1, T_2)$, iff for all arcs $uv \in A$, $u \neq s$

- 1 if $f(uv) = 2$, then $\exists wu \in A : f(wu) = 2$,
- 2 if $f(uv) = 1$, then either $\exists wu \in A : f(wu) = 1$, or $\lambda(s, u) \geq 2$, and moreover
- 3 for any receiver $t \in T_1$ with $\lambda(s, t) = 1$, there is a 1-limited arc entering t , and
- 4 for any $t \in T_2$ there is a 2-limited arc entering t .
- 5 **Field size:** $q > |T_1| + |T_2|$.

Algorithmic goal for two layers

- $\tau = (T_1, T_2)$ is a proper demand

www.leaderstudio.net

Algorithmic goal for two layers

- $\tau = (T_1, T_2)$ is a proper demand
- Goal: an algorithm constructing a network code \mathbf{c} feasible for

www.leaderstudio.net

Algorithmic goal for two layers

- $\tau = (T_1, T_2)$ is a proper demand
- Goal: an algorithm constructing a network code \mathbf{c} feasible for
- $\tau' = (T_1 \cup T_2 - T'_2, T'_2)$

www.leaderstudio.net

Algorithmic goal for two layers

- $\tau = (T_1, T_2)$ is a proper demand
- Goal: an algorithm constructing a network code \mathbf{c} feasible for
- $\tau' = (T_1 \cup T_2 - T'_2, T'_2)$
- where $T'_2 \subseteq T_2$ is the unique maximum size subset

www.leaderstudio.net

Algorithmic goal for two layers

- $\tau = (T_1, T_2)$ is a proper demand
- Goal: an algorithm constructing a network code \mathbf{c} feasible for
- $\tau' = (T_1 \cup T_2 - T'_2, T'_2)$
- where $T'_2 \subseteq T_2$ is the unique maximum size subset
- such that a feasible network code exists for τ' .

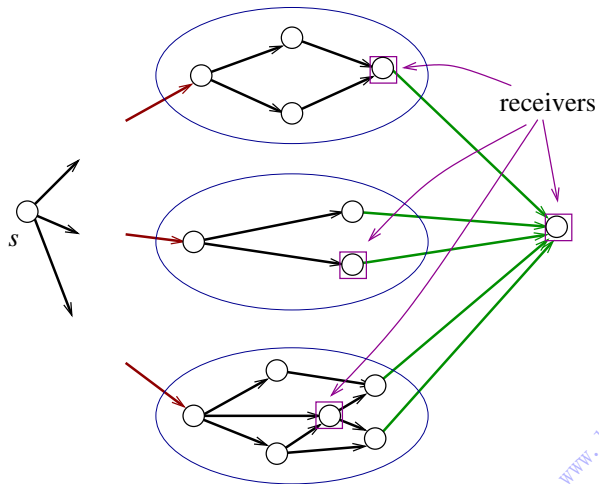
www.leaderstudio.net

Algorithmic goal for two layers

- $\tau = (T_1, T_2)$ is a proper demand
- Goal: an algorithm constructing a network code \mathbf{c} feasible for
- $\tau' = (T_1 \cup T_2 - T'_2, T'_2)$
- where $T'_2 \subseteq T_2$ is the unique maximum size subset
- such that a feasible network code exists for τ' .
- **By the theorem, it is enough to give a realizable arc-limit.**

www.leaderstudio.net

Constraints



Algorithm for two layers

- Z_i : maximal sets with in-degree 1, containing a receiver.

www.leaderstudio.net

Algorithm for two layers

- Z_i : maximal sets with in-degree 1, containing a receiver.
- $I(Z_i)$: arcs incident with Z_i .

www.leaderstudio.net

Algorithm for two layers

- Z_i : maximal sets with in-degree 1, containing a receiver.
- $I(Z_i)$: arcs incident with Z_i .
- Z : nodes not reachable from s in $D - \bigcup I(Z_i)$.

Algorithm for two layers

- Z_i : maximal sets with in-degree 1, containing a receiver.
- $I(Z_i)$: arcs incident with Z_i .
- Z : nodes not reachable from s in $D - \bigcup I(Z_i)$.
- $T'_2 := T_2 - Z$.

Algorithm for two layers

- Z_i : maximal sets with in-degree 1, containing a receiver.
- $I(Z_i)$: arcs incident with Z_i .
- Z : nodes not reachable from s in $D - \bigcup I(Z_i)$.
- $T'_2 := T_2 - Z$.
- $f(a) := 1$, if $a \in I(Z)$, and 2 otherwise.

Algorithm for two layers

- Z_i : maximal sets with in-degree 1, containing a receiver.
- $I(Z_i)$: arcs incident with Z_i .
- Z : nodes not reachable from s in $D - \bigcup I(Z_i)$.
- $T'_2 := T_2 - Z$.
- $f(a) := 1$, if $a \in I(Z)$, and 2 otherwise.
- Feasibility is easy to prove,

Algorithm for two layers

- Z_i : maximal sets with in-degree 1, containing a receiver.
- $I(Z_i)$: arcs incident with Z_i .
- Z : nodes not reachable from s in $D - \bigcup I(Z_i)$.
- $T'_2 := T_2 - Z$.
- $f(a) := 1$, if $a \in I(Z)$, and 2 otherwise.
- Feasibility is easy to prove,
- and linear time algorithm is also easy.

- For 3 layers (can be extended for k).

Goals for pre-limiting algorithm

- For 3 layers (can be extended for k).
- Goals: in linear time and in a distributed way

www.leaderstudio.net

Goals for pre-limiting algorithm

- For 3 layers (can be extended for k).
- Goals: in linear time and in a distributed way
- determine $\lambda(s, v)$, but if ≥ 3 , this fact is enough,

www.leaderstudio.net

Goals for pre-limiting algorithm

- For 3 layers (can be extended for k).
- Goals: in linear time and in a distributed way
- determine $\lambda(s, v)$, but if ≥ 3 , this fact is enough,
- if $\lambda(s, v) = 1$, determine the incoming arc of the maximal \bar{sv} in-degree 1 set,

www.leaderstudio.net

Goals for pre-limiting algorithm

- For 3 layers (can be extended for k).
- Goals: in linear time and in a distributed way
- determine $\lambda(s, v)$, but if ≥ 3 , this fact is enough,
- if $\lambda(s, v) = 1$, determine the incoming arc of the maximal $\bar{s}v$ in-degree 1 set,
- if $\lambda(s, v) = 2$, determine the incoming pair of arcs of the maximal $\bar{s}v$ in-degree 2 set.

www.leaderstudio.net

- Sketch: s sends $(*, *, *)$ on every outgoing arc.

Distributed pre-limiting algorithm

- Sketch: s sends $(*, *, *)$ on every outgoing arc.
- v waits until hears messages on all incoming arcs,

www.leaderstudio.net

Distributed pre-limiting algorithm

- Sketch: s sends $(*, *, *)$ on every outgoing arc.
- v waits until hears messages on all incoming arcs,
- in every message it replaces $*$ by the ID of the arc it arrives.

www.leaderstudio.net

Distributed pre-limiting algorithm

- Sketch: s sends $(*, *, *)$ on every outgoing arc.
- v waits until hears messages on all incoming arcs,
- in every message it replaces $*$ by the ID of the arc it arrives.
- Next it determines $\lambda(s, v)$ and the incoming arcs of the min-cut, and sends out $(m_1(v), m_2(v), m_3(v))$ on every outgoing arc.

www.leaderstudio.net

Distributed pre-limiting algorithm

- Sketch: s sends $(*, *, *)$ on every outgoing arc.
- v waits until hears messages on all incoming arcs,
- in every message it replaces $*$ by the ID of the arc it arrives.
- Next it determines $\lambda(s, v)$ and the incoming arcs of the min-cut, and sends out $(m_1(v), m_2(v), m_3(v))$ on every outgoing arc.
- Where $m_1(v)$ is the entering arc of the 1-cut, if $\lambda(s, v) = 1$ and $*$ otherwise, and

www.leaderstudio.net

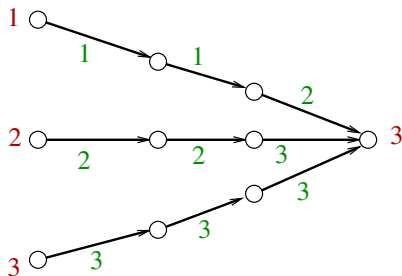
Distributed pre-limiting algorithm

- Sketch: s sends $(*, *, *)$ on every outgoing arc.
- v waits until hears messages on all incoming arcs,
- in every message it replaces $*$ by the ID of the arc it arrives.
- Next it determines $\lambda(s, v)$ and the incoming arcs of the min-cut, and sends out $(m_1(v), m_2(v), m_3(v))$ on every outgoing arc.
- Where $m_1(v)$ is the entering arc of the 1-cut, if $\lambda(s, v) = 1$ and $*$ otherwise, and
- $\{m_2(v), m_3(v)\}$ are entering arcs of a 2-cut, if there is one, and $\{*, *\}$ otherwise.

www.leaderstudio.net

Limits for 3 layers

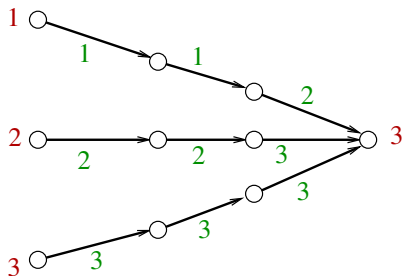
- Node extension $g : V \rightarrow \{0, 1, 2, 3\}$ of arc-limit f ,



www.leaderstudio.net

Limits for 3 layers

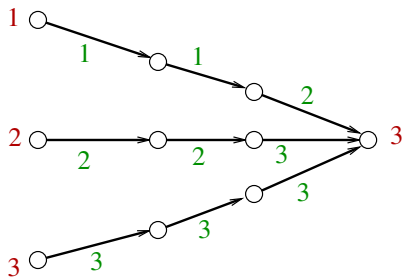
- Node extension $g : V \rightarrow \{0, 1, 2, 3\}$ of arc-limit f ,
- determined by topological order.



www.leaderstudio.net

Limits for 3 layers

- Node extension $g : V \rightarrow \{0, 1, 2, 3\}$ of arc-limit f ,
- determined by topological order.
- Defining i -fans, by an example:



www.leaderstudio.net

Theorem

Given arc-limit f and a proper demand τ , f is realizable, if for its node extension g and for every receiver t , $g(t) \geq \text{request of } t$.

Moreover the node extension of f can be determined algorithmically.

www.leaderstudio.net

Heuristic for 3 layers (sketch)

- Use distributed algorithm, write limit 1 or 2 onto the arcs determined, and 3 otherwise.

www.leaderstudio.net

Heuristic for 3 layers (sketch)

- Use distributed algorithm, write limit 1 or 2 onto the arcs determined, and 3 otherwise.
- In topological order for every node determine the node extension of f ,

www.leaderstudio.net

Heuristic for 3 layers (sketch)

- Use distributed algorithm, write limit 1 or 2 onto the arcs determined, and 3 otherwise.
- In topological order for every node determine the node extension of f ,
- if it is too small, decrease some arc-limits from 3 to 2,

www.leaderstudio.net

Heuristic for 3 layers (sketch)

- Use distributed algorithm, write limit 1 or 2 onto the arcs determined, and 3 otherwise.
- In topological order for every node determine the node extension of f ,
- if it is too small, decrease some arc-limits from 3 to 2,
- using Suurballe's algorithm with an appropriately defined cost function.

www.leaderstudio.net

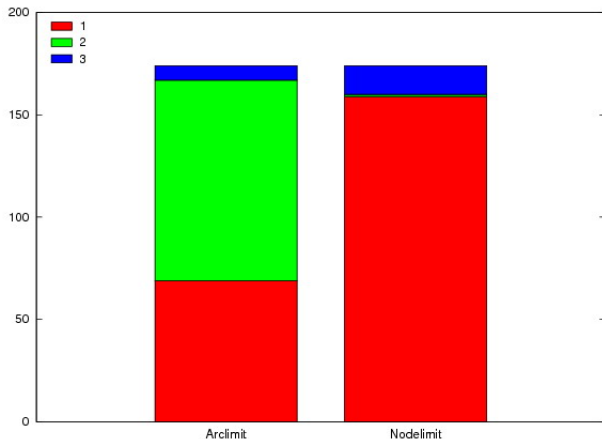
Heuristic for 3 layers (sketch)

- Use distributed algorithm, write limit 1 or 2 onto the arcs determined, and 3 otherwise.
- In topological order for every node determine the node extension of f ,
- if it is too small, decrease some arc-limits from 3 to 2,
- using Suurballe's algorithm with an appropriately defined cost function.
- **Theorem: every receiver gets the base layer, and also layer 2, if it is not 1-separated.**

www.leaderstudio.net

Comparison I

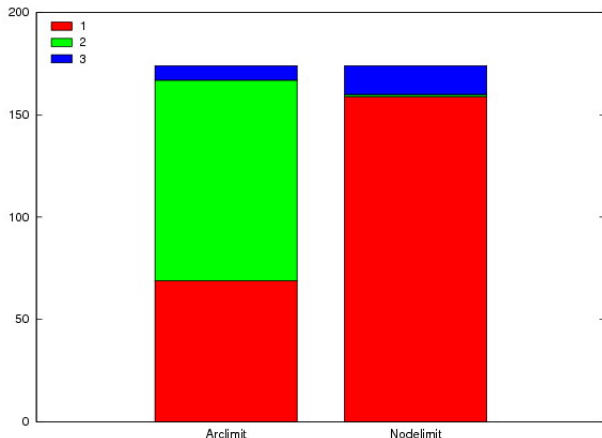
- It is hard to compare.



www.leaderstudio.net

Comparison I

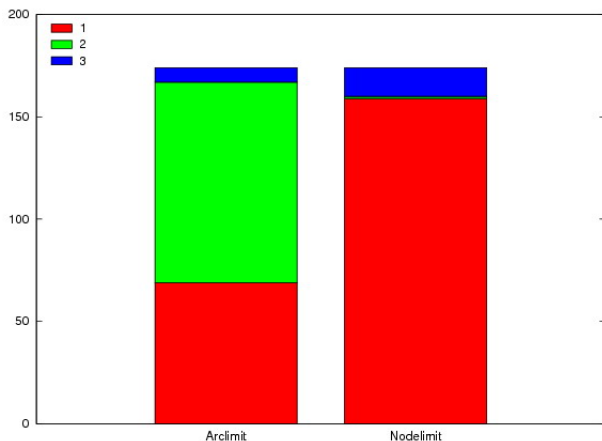
- It is hard to compare.
- What is the objective?



www.leaderstudio.net

Comparison I

- It is hard to compare.
- What is the objective?
- On approx. half of the inputs the new algorithm is clearly better:



www.leaderstudio.net

- Objective by scoring we used:

- Objective by scoring we used:
- Any receiver who gets only the base layer: 0.1 point

Comparison II

- Objective by scoring we used:
- Any receiver who gets only the base layer: 0.1 point
- A receiver with demand 2, if gets both: 2 points

www.leaderstudio.net

Comparison II

- Objective by scoring we used:
- Any receiver who gets only the base layer: 0.1 point
- A receiver with demand 2, if gets both: 2 points
- A receiver with demand 3, if gets the first two: 1.8 points

www.leaderstudio.net

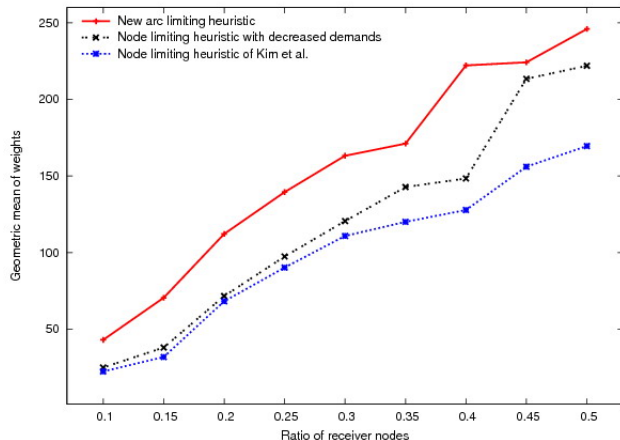
Comparison II

- Objective by scoring we used:
- Any receiver who gets only the base layer: 0.1 point
- A receiver with demand 2, if gets both: 2 points
- A receiver with demand 3, if gets the first two: 1.8 points
- A receiver with demand 3, if gets all the three: 3 points

www.leaderstudio.net

Comparison II

- 1000 nodes, 3500 arcs, each dot represents the average score along 20 runs:



www.leaderstudio.net

Thank you for your attention!

www.leaderstudio.net