A Parallel Route Assignment Algorithm for Fault-tolerant Clos Networks in OTN switches

Lingkang Wang, Tong Ye, Tony T. Lee

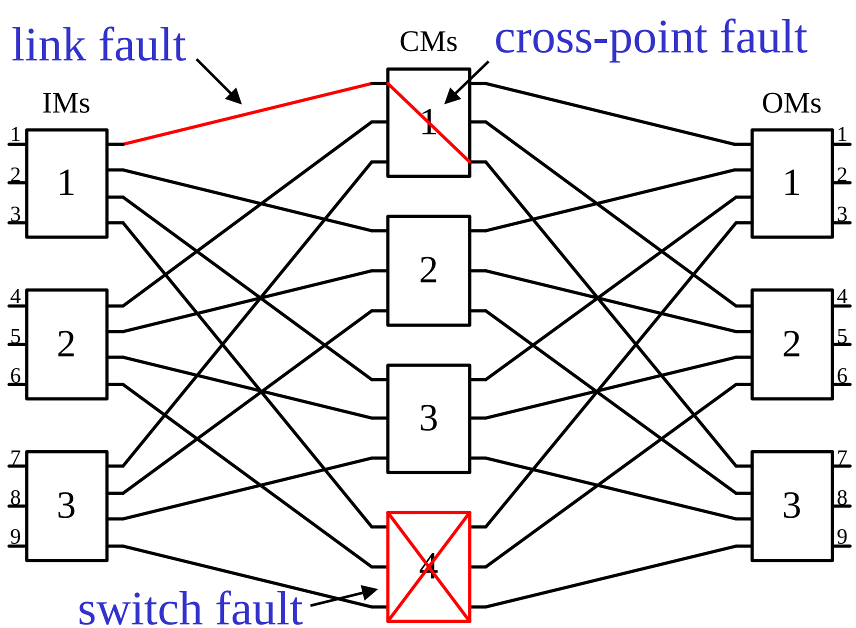
State Key Lab of Advanced Optical Communications and Networks
Shanghai Jiao Tong University

Outline

- Introduction and Overview
- Route Assignment and Complex Coloring
- Parallel Complex Coloring with Redundant Colors
- Parallel Routing Algorithm
- Conclusion

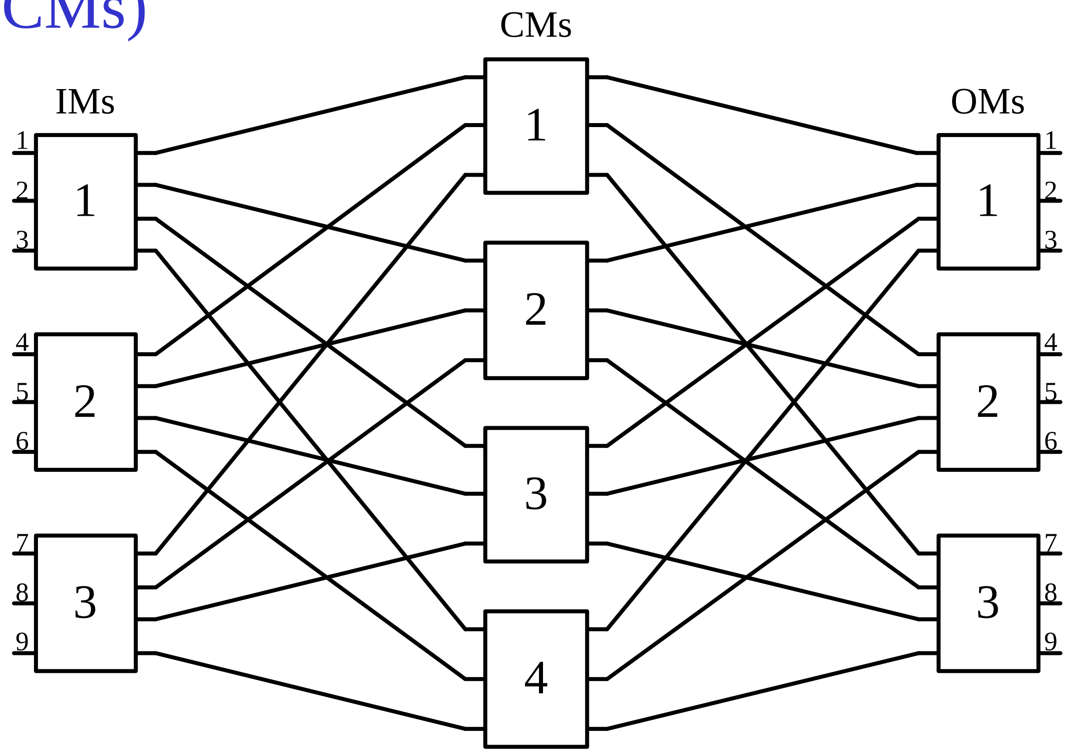
Fault tolerance

- Fault tolerance is indispensable to current switching networks.
- Fault models for switching networks
  - Link fault & Cross-point fault
    - It's two costly to detect and locate faults of such classes.
    - The corresponding re-routing algorithms are complex and manageable.
  - Switch fault
    - Effects of link faults or cross-point faults can be subsumed by effects of switch fault.



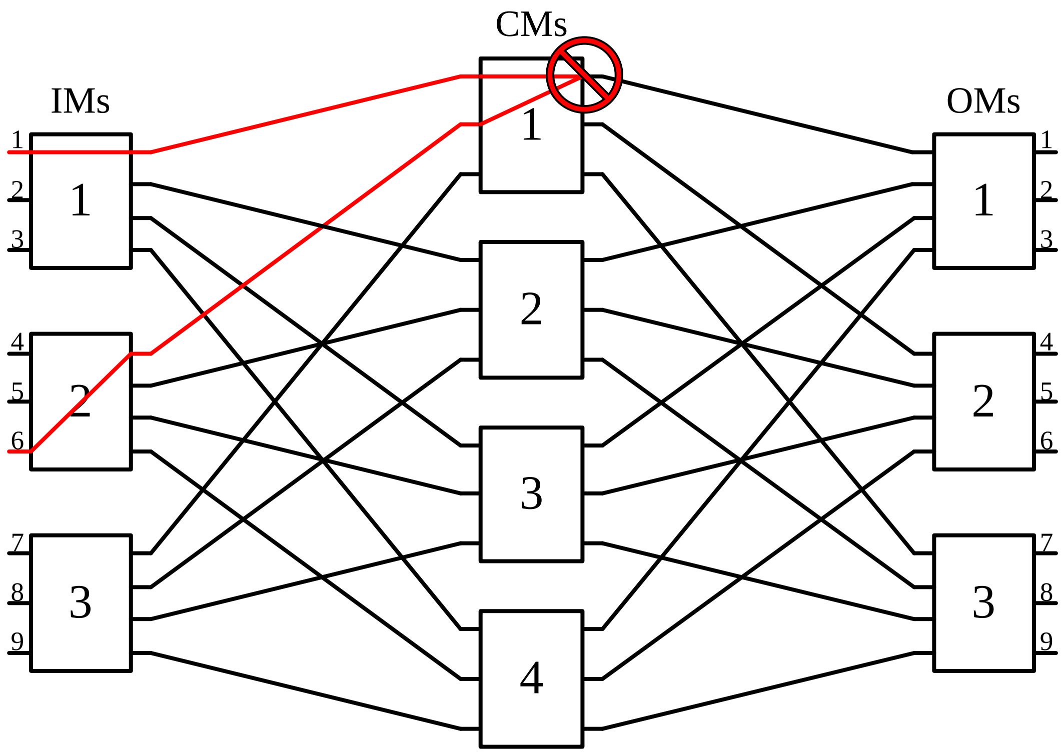
Fault-tolerant Clos Networks

- A three-stage Clos network $C(m, n, r)$
  - r $n \times n$ input modules, m $r \times r$ central modules and r $n \times n$ outputs modules
  - Rearrangeable non-blocking condition: $m \geq n$
- Clos networks can be made fault-tolerant with extra central modules (CMs)



Route Assignment[1]

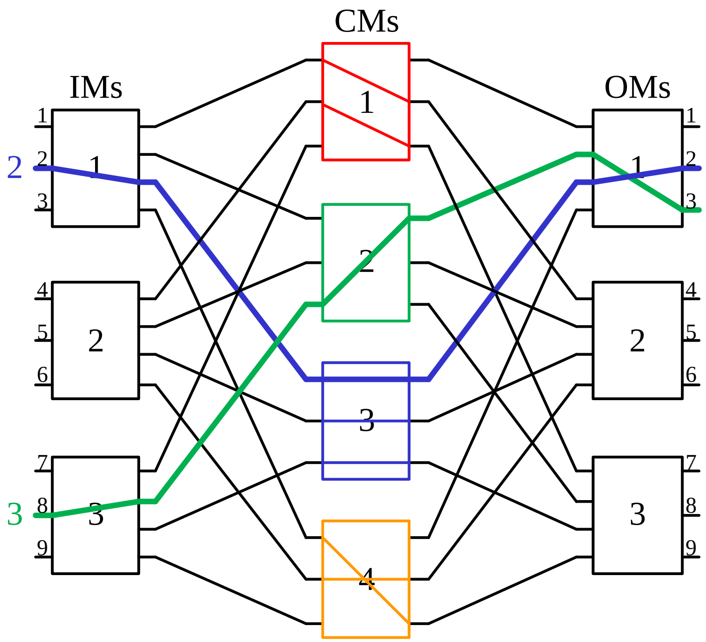
- Route assignment is necessary for assigning internally conflict-free paths in three-stage fault-tolerant Clos networks



[1] H. J. Chao, Z. Jing, and S. Y. Liew, *IEEE Commun. Mag.*, vol. 41, no. 10, pp. 46–54, Oct. 2003.

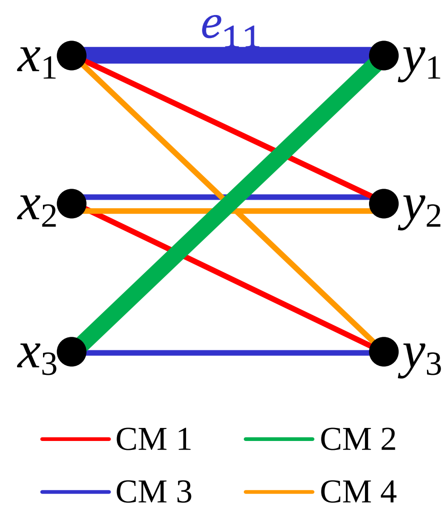
Bipartite Graph Model

- Route assignment problem in three-stage Clos networks can be formulated as the bipartite-graph edge-coloring problem



(a) Fault-tolerant Clos network

$$\pi = \begin{pmatrix} 1 & \boxed{2} & 3 & 4 & 5 & 6 & 7 & \boxed{8} & 9 \\ 5 & \boxed{2} & 7 & 8 & 6 & 4 & 9 & \boxed{3} & x \end{pmatrix}$$



(b) Bipartite graph model

Vertex x_i (y_j): input (output) module i (j)
Edge e_{ij}: request from x_i to y_j

Current Routing Algorithms

- *GenericRearrangementRouting*[1]
  - Link or cross-point fault model
  - High complexity: $O(|LC|\log N + N^{1.5})$, where $N = nk$ and $|LC|$ is the total number of faulty cross-points
- Decomposition algorithm[2,3]
  - Complexity: $O(nr^2)$ for $C(m, n, r)$
  - It shows that the running time can be significantly improved as the number of extra switch modules increases

[1] Y. Yang, J. Wang, "A fault-tolerant rearrangeable permutation network," *IEEE Trans. Comput.*, vol. 53, no. 4, pp. 414-426, Apr. 2004.[2] M. Karol and C-L. I, *IEEE Trans. Commun.*, vol. 40, no. 2, pp. 431–439, Feb. 1992.

[2] H. Y. Lee, F. K. Hwang, J. D. Carpinelli, "A new decomposition algorithm for rearrangeable Clos interconnection networks," *IEEE Trans. Commun.*, vol. 44, no. 11, pp. 1572-1578, Nov. 1996.

[3] H. Y. Lee and J. D. Carpinelli, "Routing algorithms in fault tolerant Clos networks," in *Conf. Inform. Sci. Syst.*, Princeton University, NJ, Mar. 1994, pp. 227-231.

[4] N. Das, J. Dattagupta, "Two-pass rearrangeability in faulty Benes networks," *Journal of Parallel and Distributed Computing*, 1996, 35(2): 191-198.

Our Work

- A parallel routing algorithm for three-stage Clos network
  - Low complexity: $O(\frac{\sqrt{N}(m-1)}{m-1+(m-\sqrt{N})\log N}\log N)$ (m: #of central modules)
  - Fault tolerance: a quick recovery from switch module failures

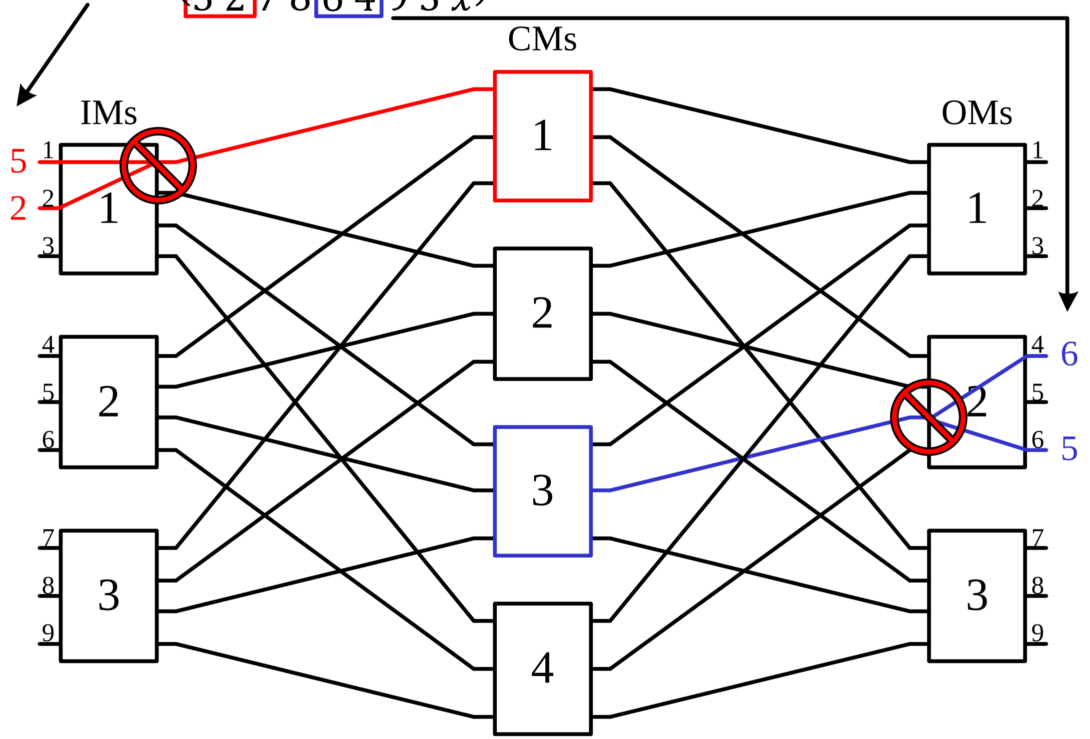
Outline

- Introduction and Overview
- Route Assignment and Complex Coloring
  - Route Assignment in fault-tolerant Clos networks
  - Parallel Complex Coloring of Bipartite Graph
    - Rearrangeability
- Parallel Complex Coloring with Redundant Colors
- Parallel Routing Algorithm
- Conclusion

Route Assignment

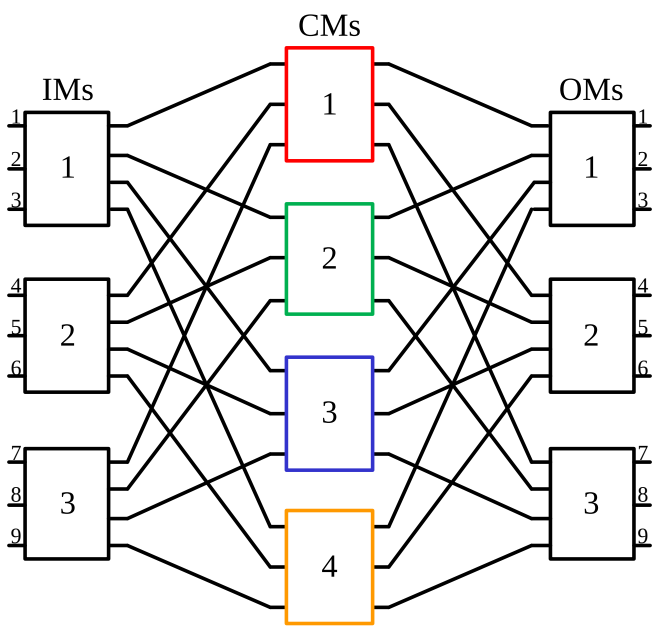
- A set of input-output ports matches is given
- Constraint
  - The path assignments of two input (output) ports on the same switching module must be distinct in order to avoid internal conflicts

$$\pi = \begin{pmatrix} 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 \\ 5 & 2 & 7 & 8 & 6 & 4 & 9 & 3 & x \end{pmatrix}$$

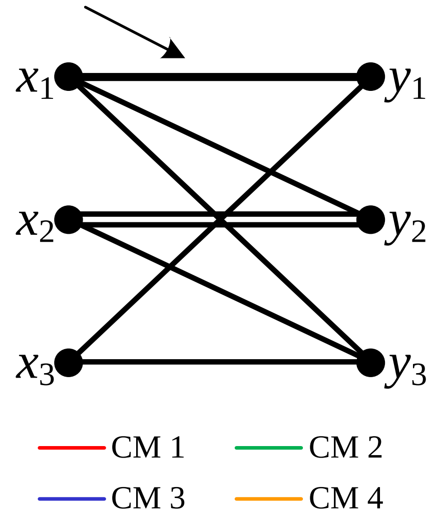


Bipartite Graph Model

- An fault-tolerant Clos network $C(m,n,r)$
  - Input/output modules ⇔ vertex set X/Y, $|X|=|Y|=r$;
  - Call requests ⇔ edge set E, $|E|=nr$;
  - Size of input/output modules ⇔ maximum degree $\Delta = n$;
  - Central modules ⇔ color set C, $|C|=m$



$$\pi = \begin{pmatrix} 1 & \boxed{2} & 3 & 4 & 5 & 6 & 7 & 8 & 9 \\ 5 & \boxed{2} & 7 & 8 & 6 & 4 & 9 & 3 & x \end{pmatrix}$$

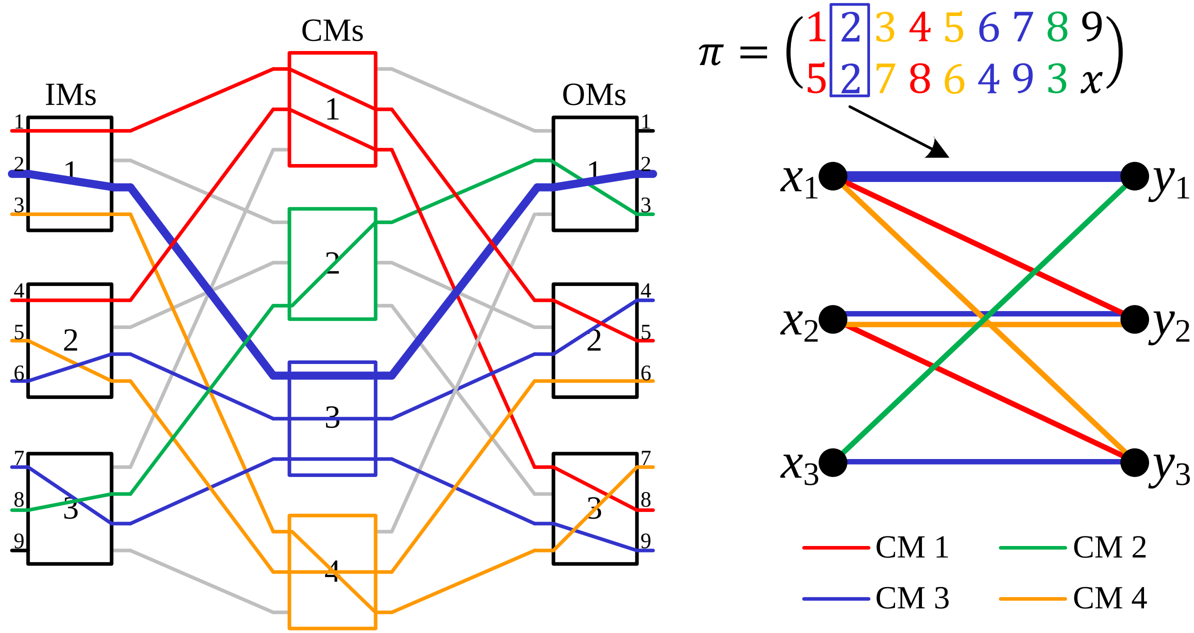


Problem Formulation

- Route assignment in fault-tolerant Clos networks

⇔ Edge coloring problem of bipartite graphs with extra colors.

  - Coloring a Δ-edge-colorable bipartite graph with $\Delta + \delta$ colors, where $\Delta = n, \delta = m - n > 0$

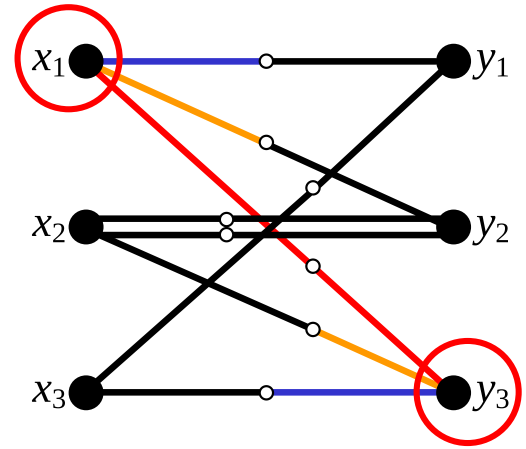


Outline

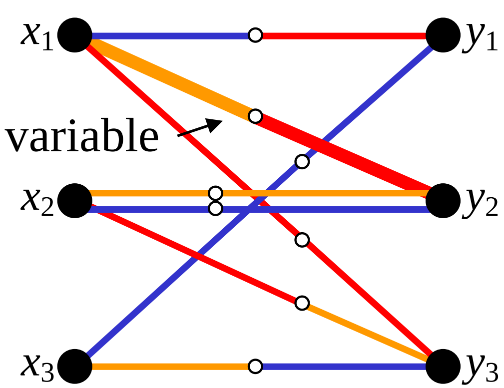
- Introduction and Overview
- Route Assignment and Complex Coloring
  - Route Assignment in RNB Clos networks
  - Parallel Complex Coloring of Bipartite Graph
    - Rearrangeability
- Parallel Complex Coloring with Redundant Colors
- Parallel Routing Algorithm
- Conclusion

Edge Coloring Constraints

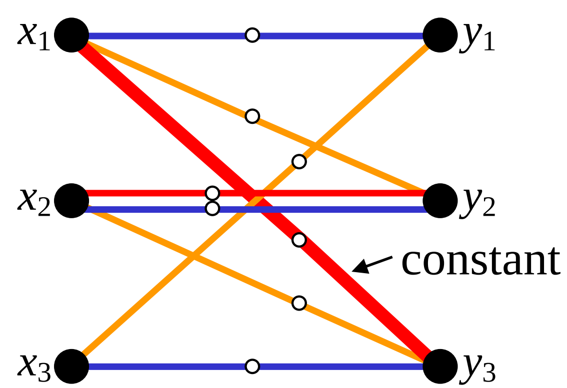
- Vertex constraint
  - Colors assigned to links incident to the same vertex are all distinct



- Edge constraint
  - Variable-colored edge
  - Constant-colored edge



Consistent coloring of G

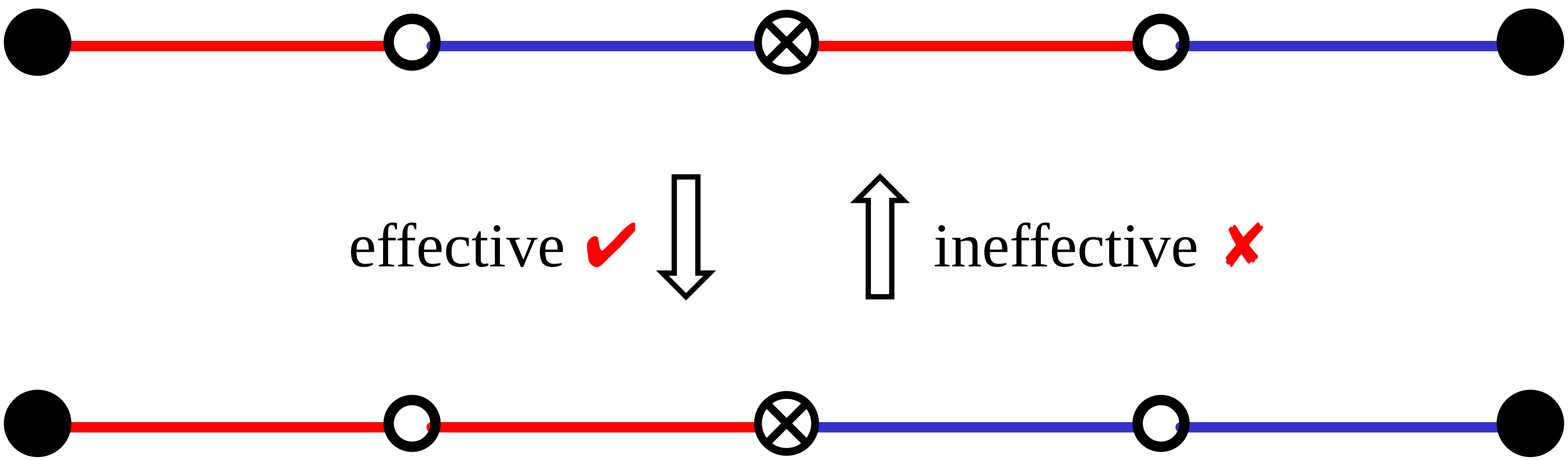


Proper coloring of G

[1] T. Lee, Y. Wan, and H. Guan, Int. J. Comput. Math. 90 (2013), 228–245.

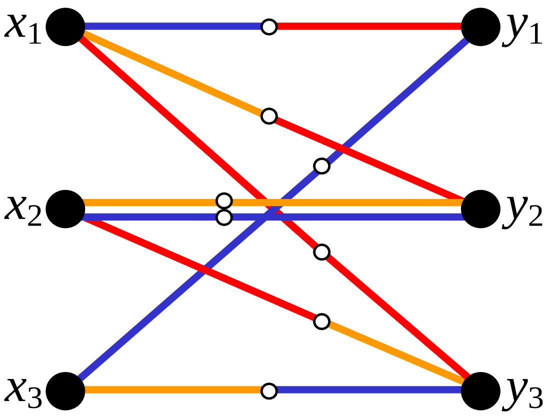
Color-Exchange Operation

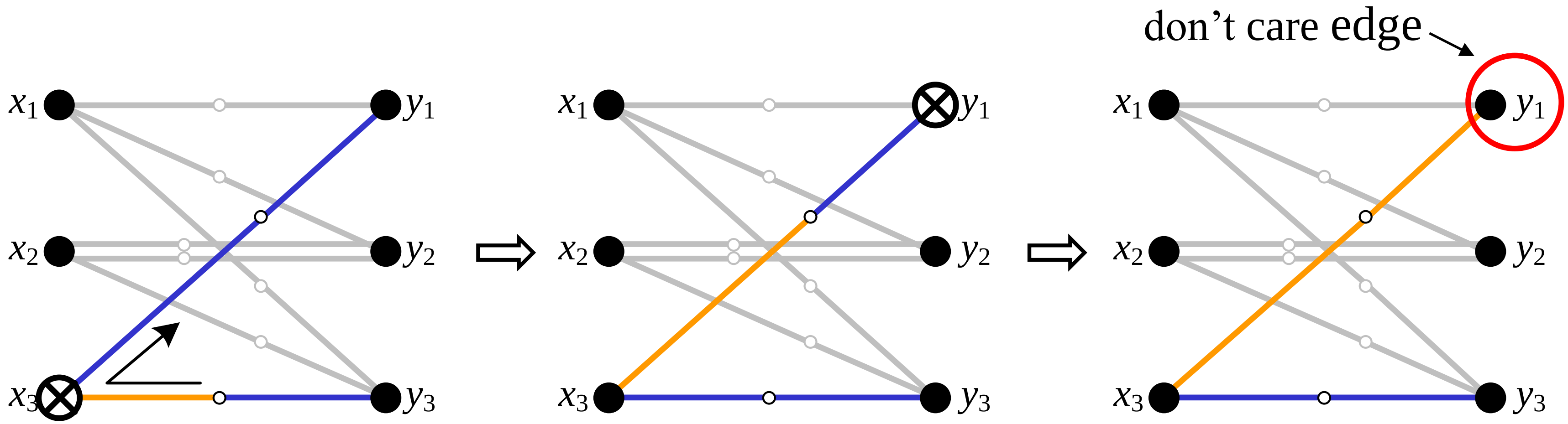
- Color-exchange operations preserve the consistency of vertex constraint
- A color-exchange operation is effective if it does not increase the number of variables



(a, b) Subgraph

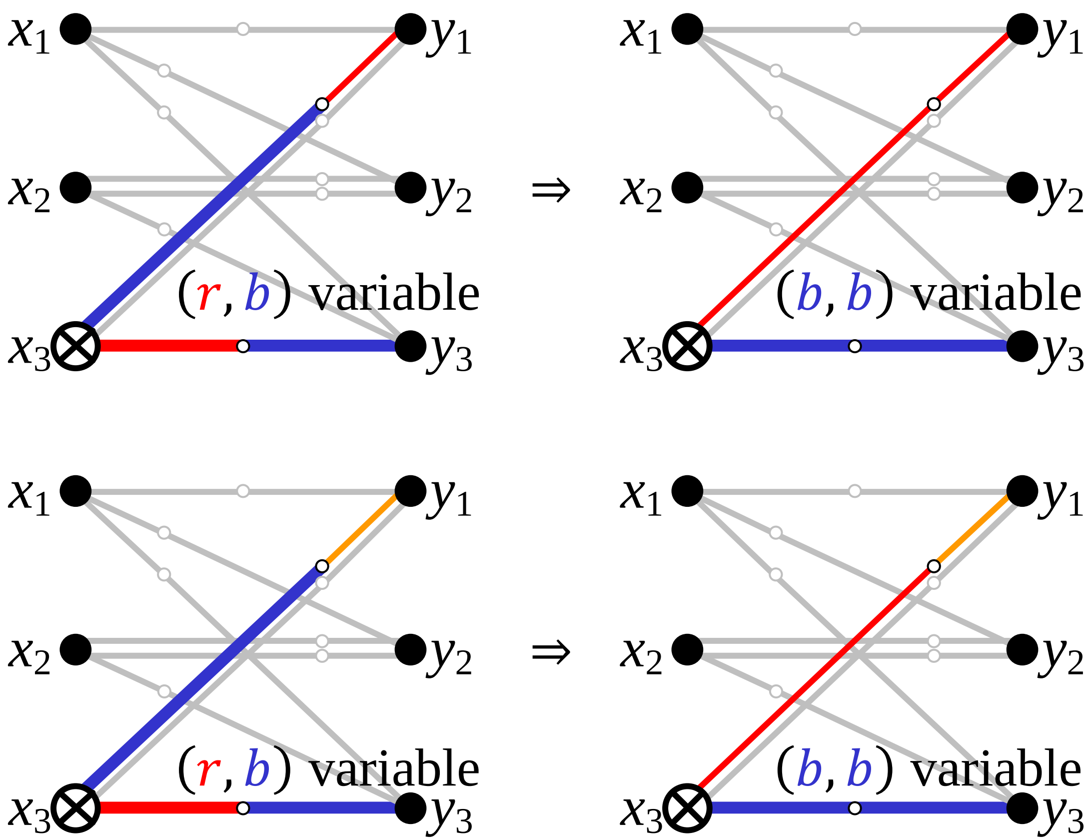
- A (a, b) variable is only allowed to move within a two-colored (a, b) subgraph
  - Don't care edge: An (a, b) open path may terminate on such a vertex without a or b link





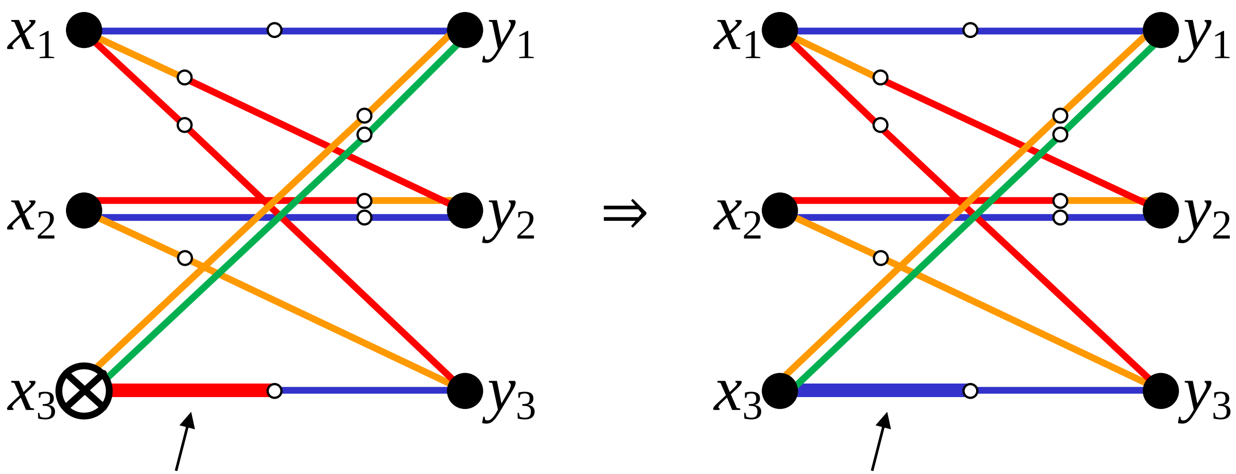
Variable Elimination: Hitting Variable

- More and more difficult to hit other variables along with the elimination process



Variable Elimination: Hitting Don’t Care Edge

- Number of don’t care edges keeps unchanged in whole elimination process
- Redundant colors speed up variable elimination

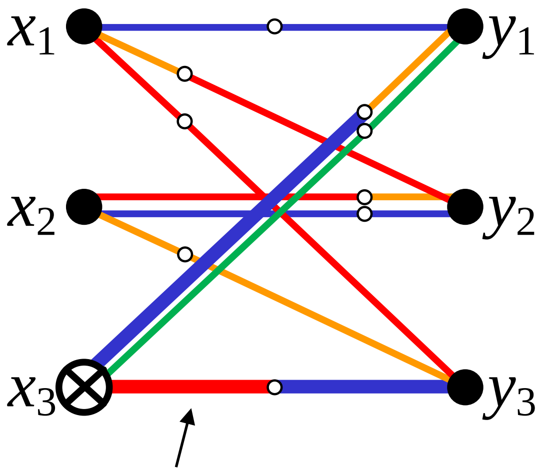


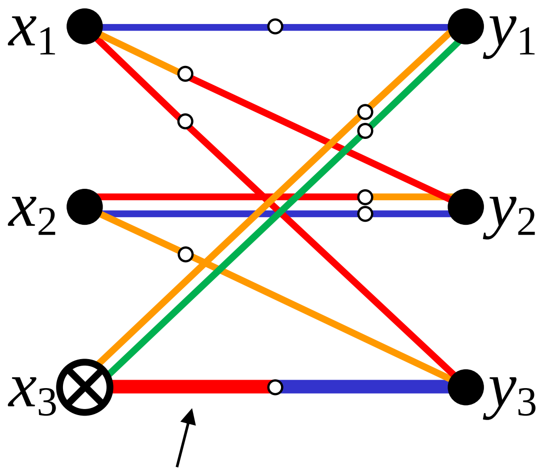
don’t care edge without color blue

replace color red with color blue

Don’t Care vs. Variable

- Two kinds of elimination are exclusive





x_3 uses color blue

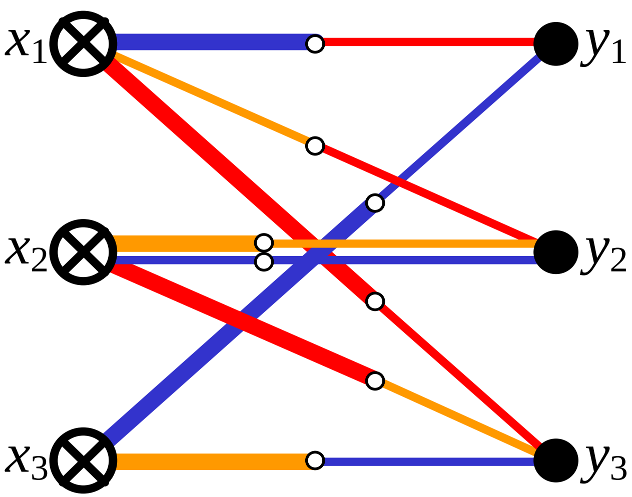
$\Rightarrow$ Eliminated by hitting $(b,*)$ variable

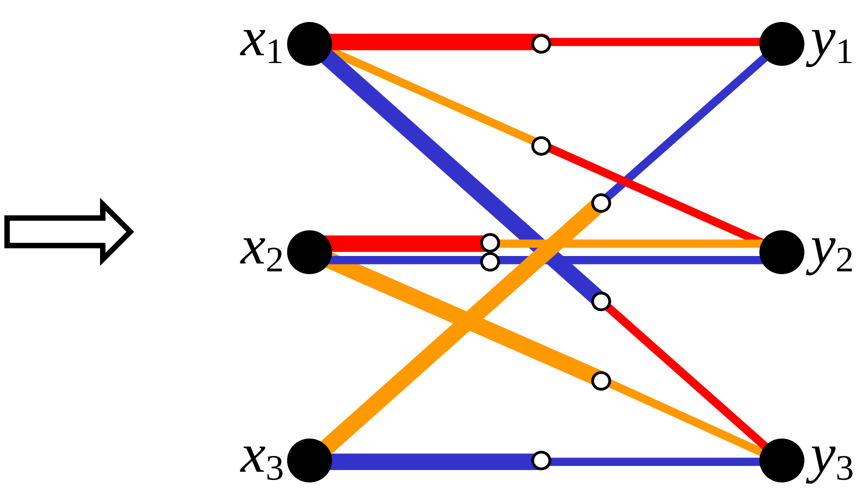
x_3 **DO NOT** use color blue

$\Rightarrow$ Eliminated by a don’t care edge

Parallel Complex Coloring

- Variables can be eliminated by color-exchange simultaneously
  - The effectiveness still holds when color exchange operations are simultaneously performed on two non-adjacent vertices
  - **High efficiency of variable eliminations!**



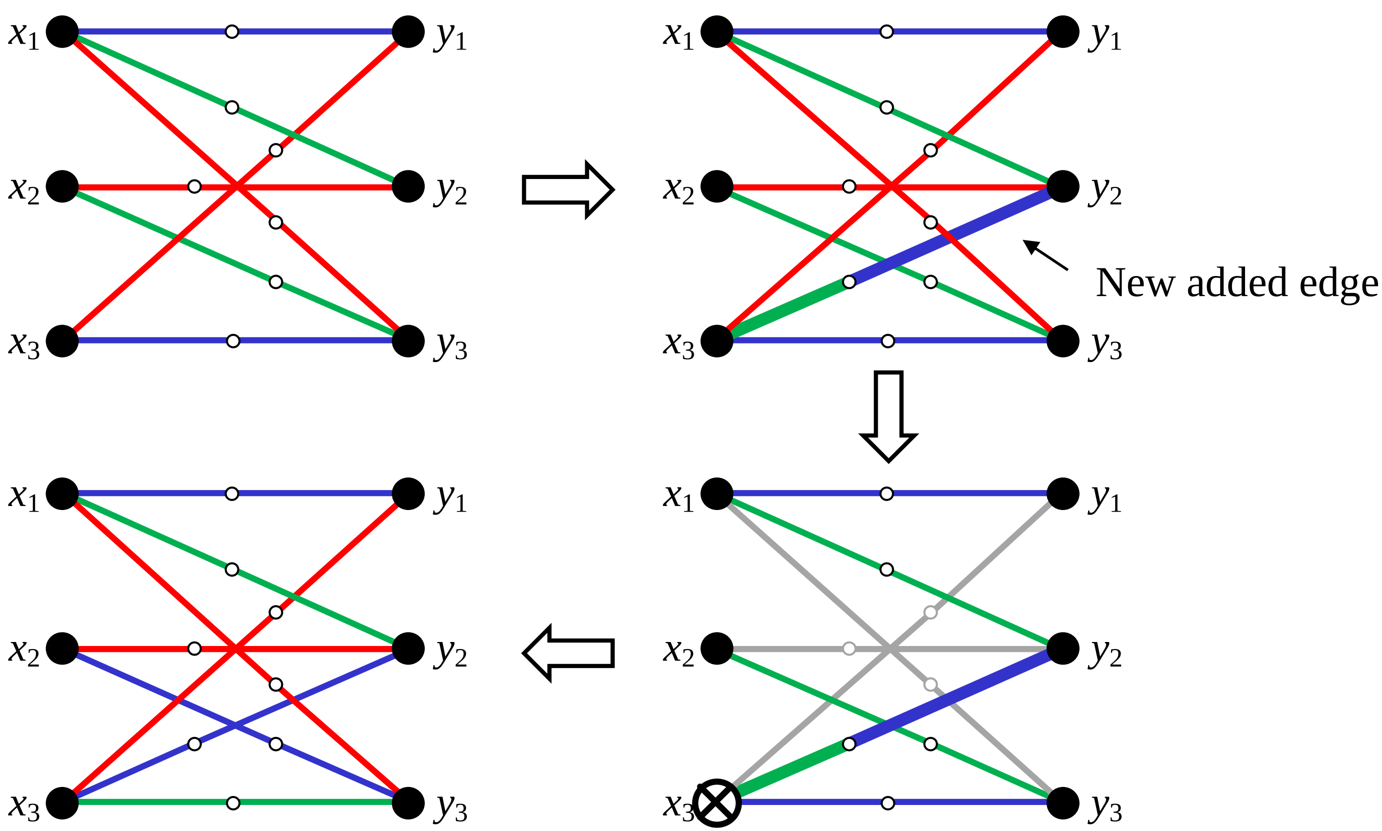


Outline

- Introduction and Overview
- Route Assignment and Complex Coloring
  - Route Assignment in RNB Clos networks
  - Parallel Complex Coloring of Bipartite Graph
    - Rearrangeability
- Parallel Complex Coloring with Redundant Colors
- Parallel Routing Algorithm
- Conclusion

Rearrangeability

- When the graph is slightly changed, only partial changes of the existing coloring are needed

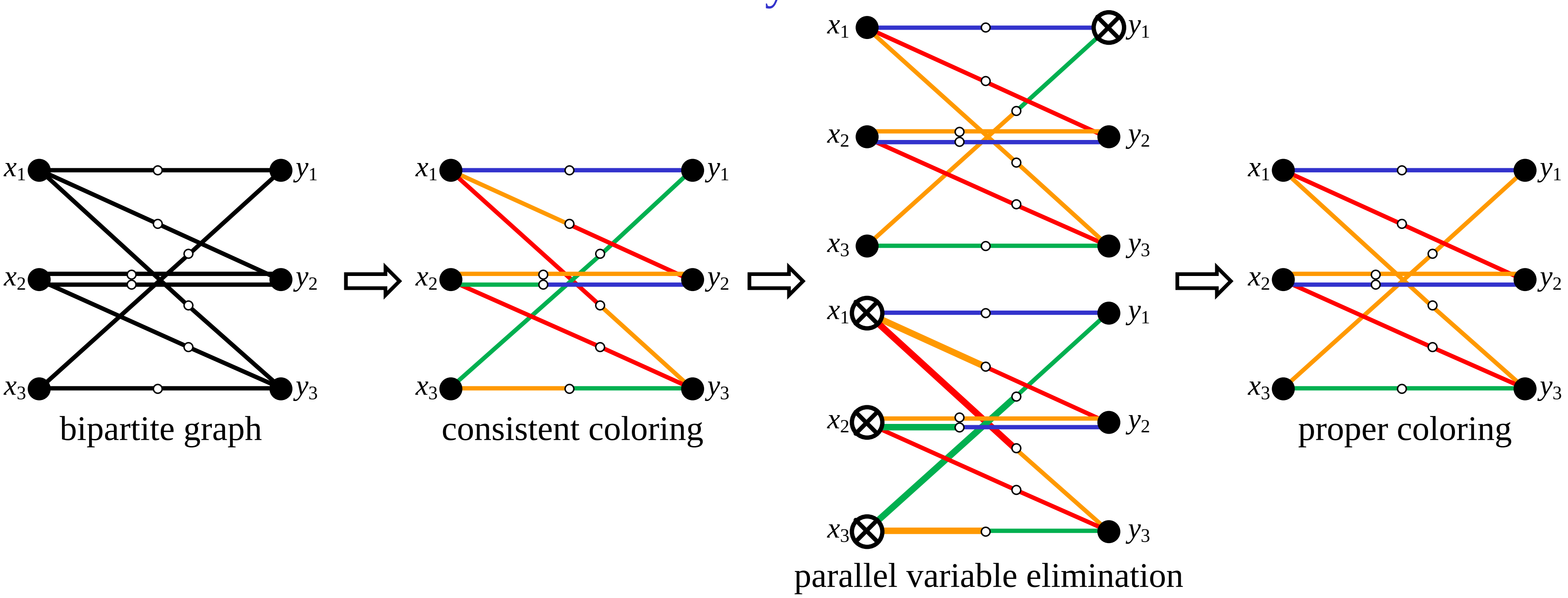


Outline

- Introduction and Overview
- Route Assignment and Complex Coloring
- Parallel Complex Coloring with Redundant Colors
  - Deadlock Variables
  - Stopping Rule
- Parallel Routing Algorithm
- Conclusion

Parallel Complex Coloring with Extra Colors

- Graph Initialization
  - For each vertex, choose a random color out of $\Delta + \delta$ colors in C for each of its associated links
    - An example: $\Delta = 3, \delta = 1$
- Parallel Complex Coloring
  - For $G = (X \cup Y, E)$, simultaneous color exchanges are performed on vertices in X and Y alternatively

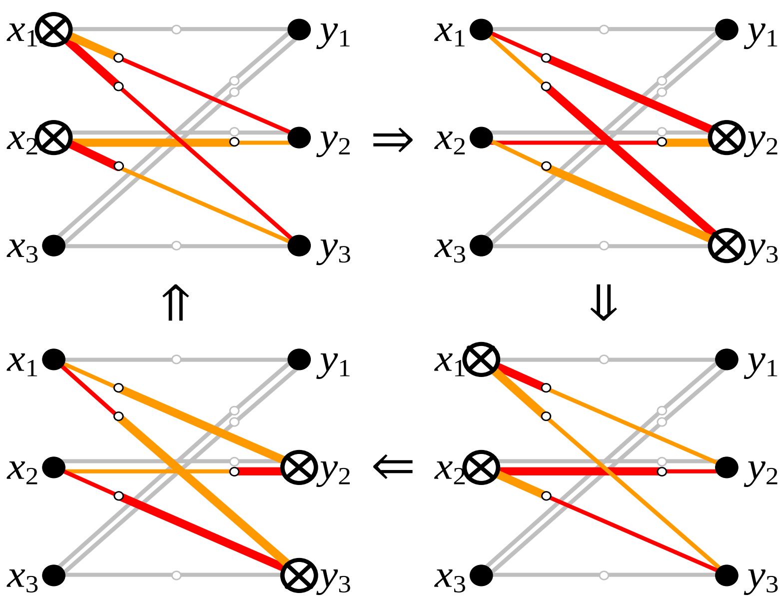


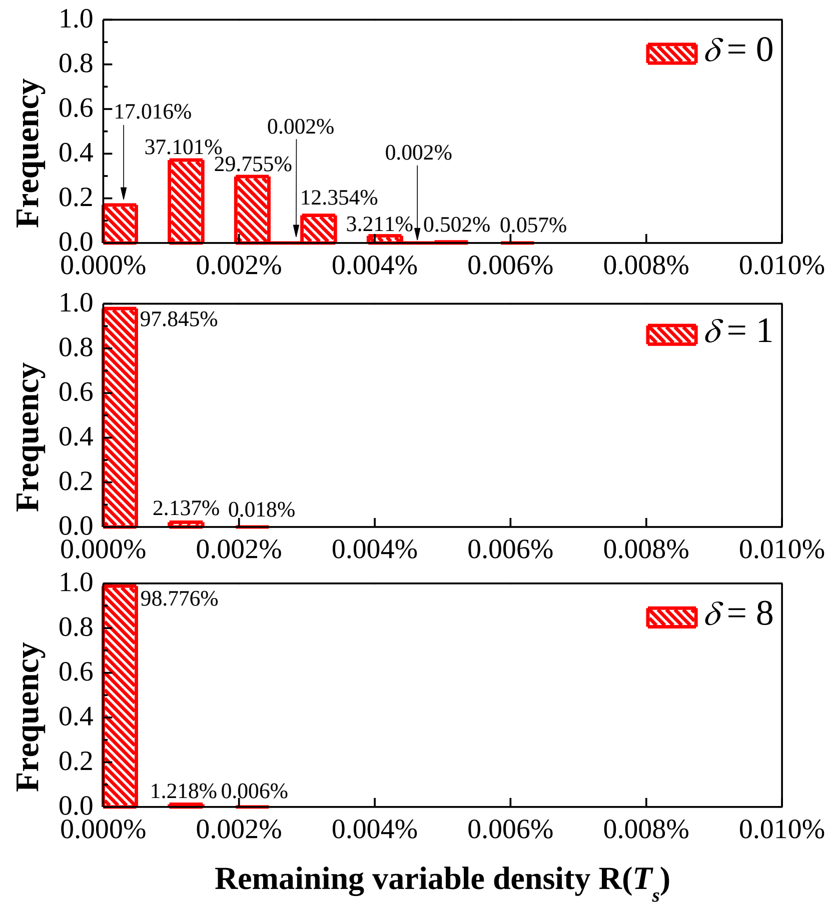
Outline

- Introduction and Overview
- Preliminaries of Routing and Complex Coloring
- Parallel Complex Coloring with Extra Colors
  - Deadlock Variables
  - Stopping Rule
- Routing Algorithm for Fault-tolerant Clos Networks
- Conclusion

Deadlock Variables

- Variables may be trapped in an infinite loop
  - Eliminated by sequential color exchange
- Redundant colors reduce deadlock situation





Outline

- Introduction and Overview
- Preliminaries of Routing and Complex Coloring
- Parallel Complex Coloring with Extra Colors
  - Deadlock Variables
  - Stopping Rule
- Routing Algorithm for Fault-tolerant Clos Networks
- Conclusion

Notation

- Variable density

$$R(t) = \frac{\text{\# of variables}}{\text{\# of edges}} \text{ (after } t \text{ iterations)}$$

- Variable elimination rate

$$\alpha(t) = \frac{\text{\# of eliminated variables}}{\text{\# of variables}} \text{ (of } t^{th} \text{ iteration)}$$

- Hitting time $h(t)$
  - Expected number of iterations needed for a variable to hit another variable of t^{th} iteration
  - $h(t) \propto 1/\alpha(t)$

Elimination Process

Suppose $\alpha(t) = \alpha$ and $h(t) = 1/\alpha$.

$$\underbrace{|E|R(t)}_{\substack{\#\, of\ variables \\ after\ t\ iterations}} - \underbrace{|E|R(t+1)}_{\substack{\#\, of\ variables \\ after\ (t+1)\ iterations}} = \underbrace{\alpha|E|R(t)}_{\substack{\#\, of\ eliminated\ variables \\ of\ (t+1)^{th}\ iteration}}$$

$$\Longrightarrow R(t) = (1-\alpha)^t R(0).$$

For $0 < \epsilon \ll 1$, the required number of iterations T is given by

$$(1-\alpha)^T R(0) = \epsilon.$$

For $\alpha \ll 1$,

$$T = \frac{1}{\alpha} \ln \frac{R(0)}{\epsilon} = \frac{h}{a} \ln \frac{R(0)}{\epsilon}.$$

Elimination Rate α / Hitting Time h

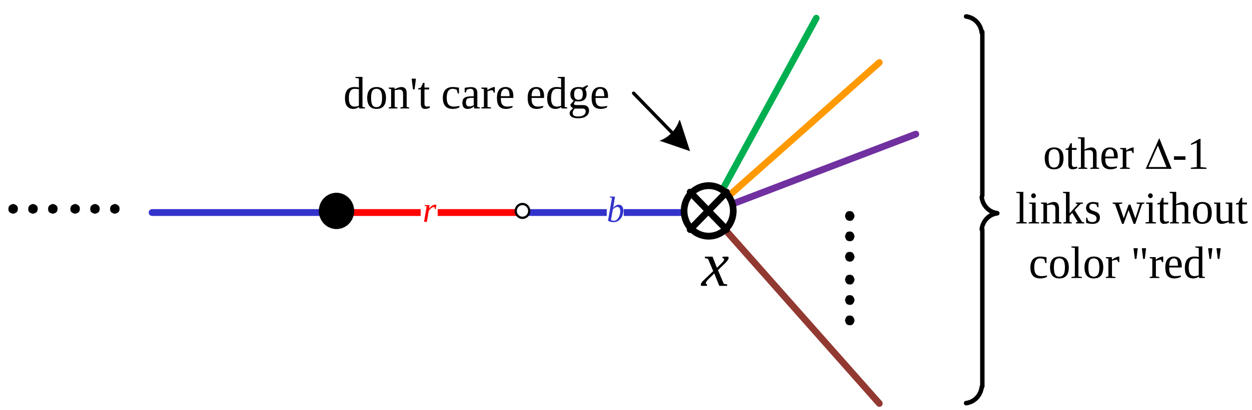
- When $\delta = 0$, hitting time h is on the order of $O(\log|V|)$[1].
  - Elimination rate α is on the order of $O(1/\log|V|)$
- When $\delta > 0$, hitting time h relates to $|V|$ as well as δ
- Difference
  - With redundant colors, don't care edges spread around the bipartite graph which greatly speed up the variable elimination process

[1] A. Fronczak, P. Fronczak, and J. Hołyst, *Phys. Rev. E*, 70(5), 2004.

Don’t care edge

- Assuming that each vertex randomly assign one of $\Delta + \delta$ colors to its associated links, the probability that a variable hits a don’t care edge in an iteration is

$$p = \frac{\binom{\Delta+\delta-2}{\Delta-1}}{\binom{\Delta+\delta-1}{\Delta-1}} = \frac{\delta}{\Delta+\delta-1}$$



Elimination Rate α / Hitting Time h

- Hitting a variable or a don't care edge are mutually exclusive
- The effect of don't care edges for elimination process can be added directly
- Thus, when $x > 0$, the elimination rate α is on the order of $O(1/\log|V| + p)$

Elimination Rate α / Hitting Time h

- Specifically,

$$\alpha = \frac{1}{a\log(|V| + b) + c} + p.$$

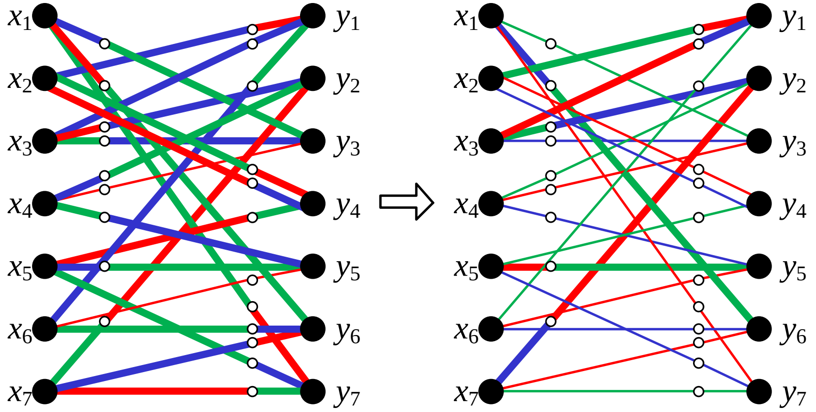
where a, b, c are constants

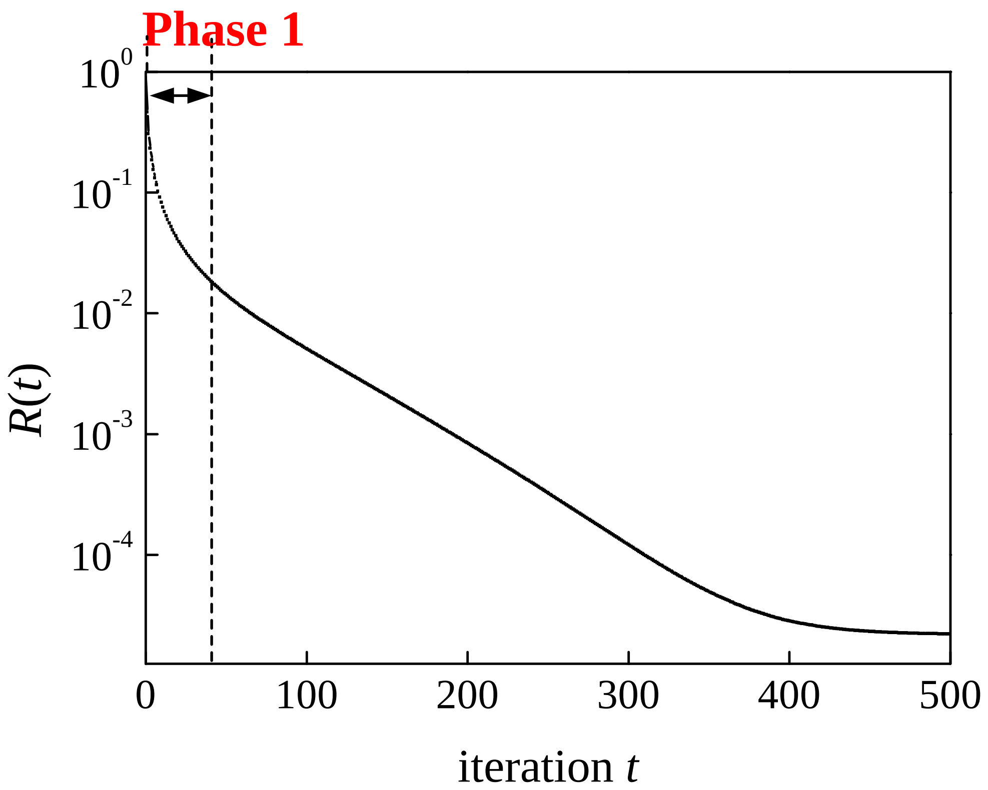
- Thus,

$$h = \frac{1}{\alpha} = \frac{1}{\frac{1}{a\log(|V| + b) + c} + p} = \frac{a\log(|V| + b) + c}{1 + p \times a\log(|V| + b) + c}$$

Elimination Process: Phase 1

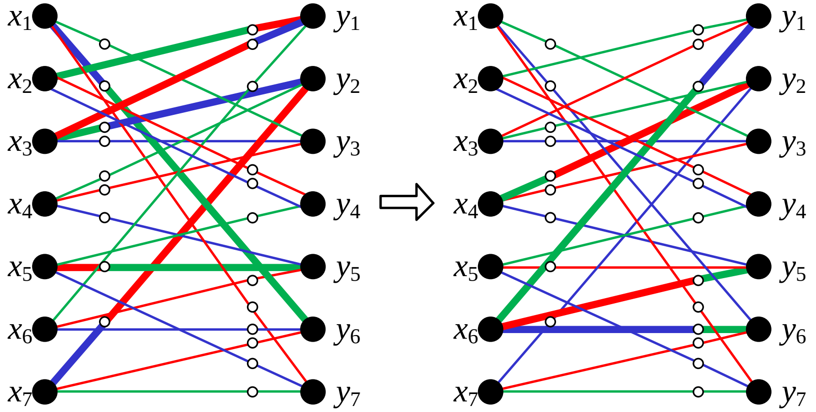
- Variable density $R(t)$

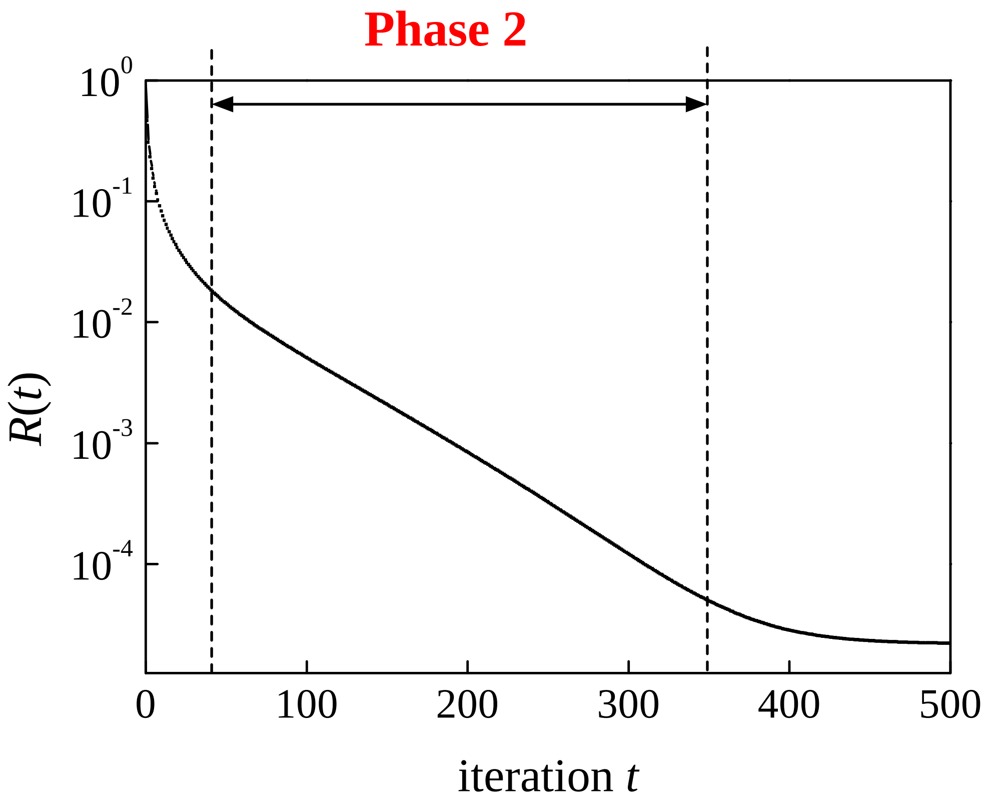




Elimination Process: Phase 2

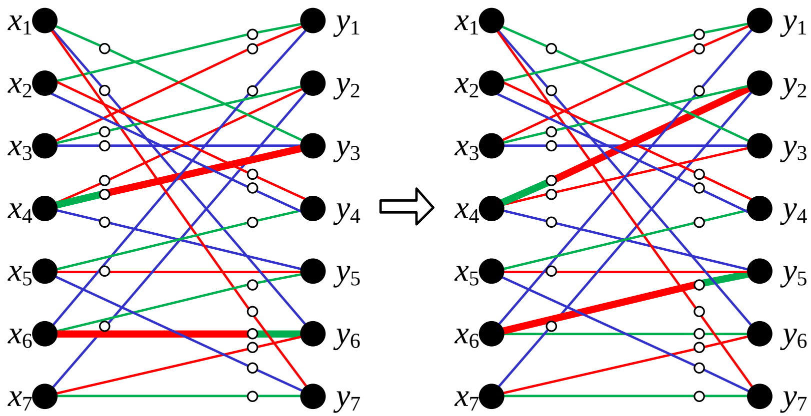
- Variable density $R(t)$

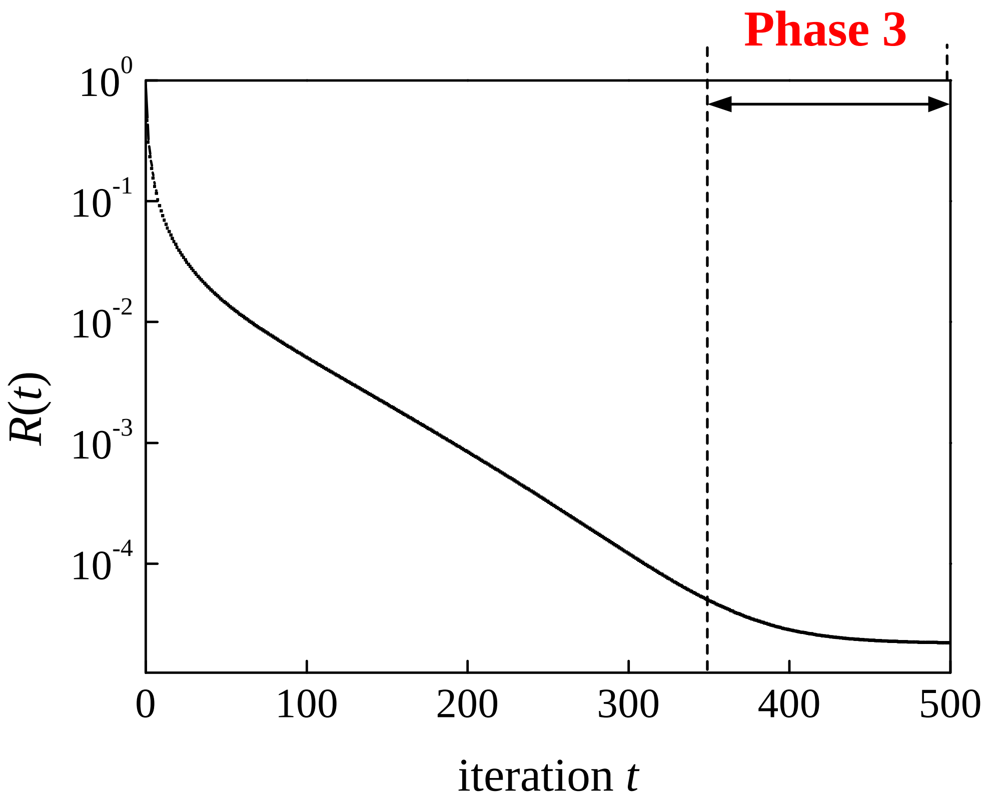




Elimination Process: Phase 3

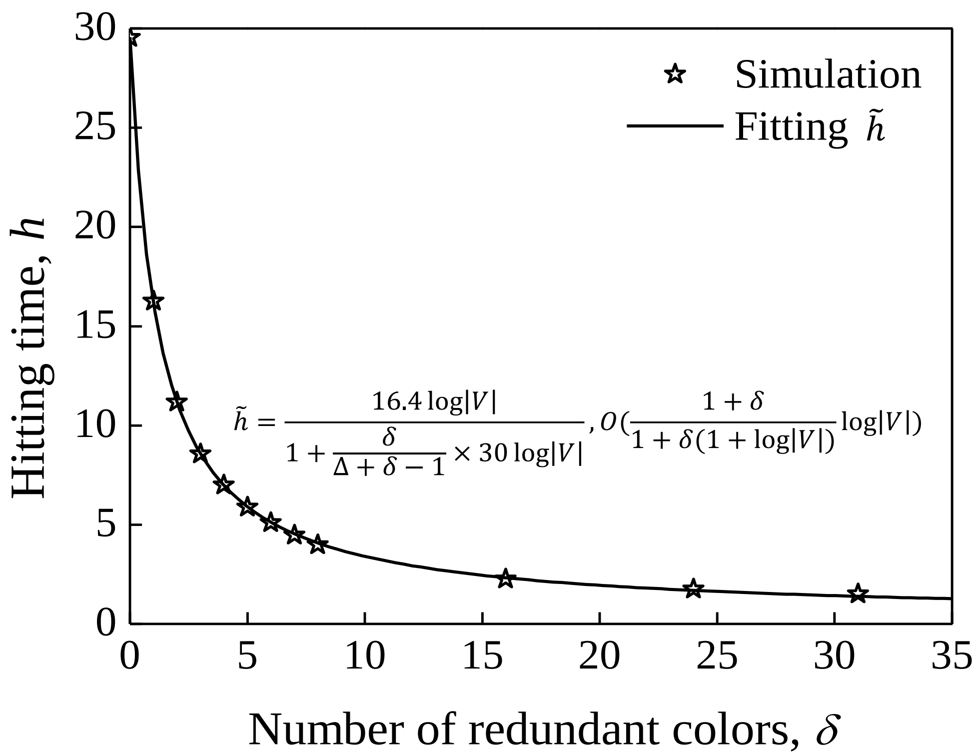
- Variable density $R(t)$





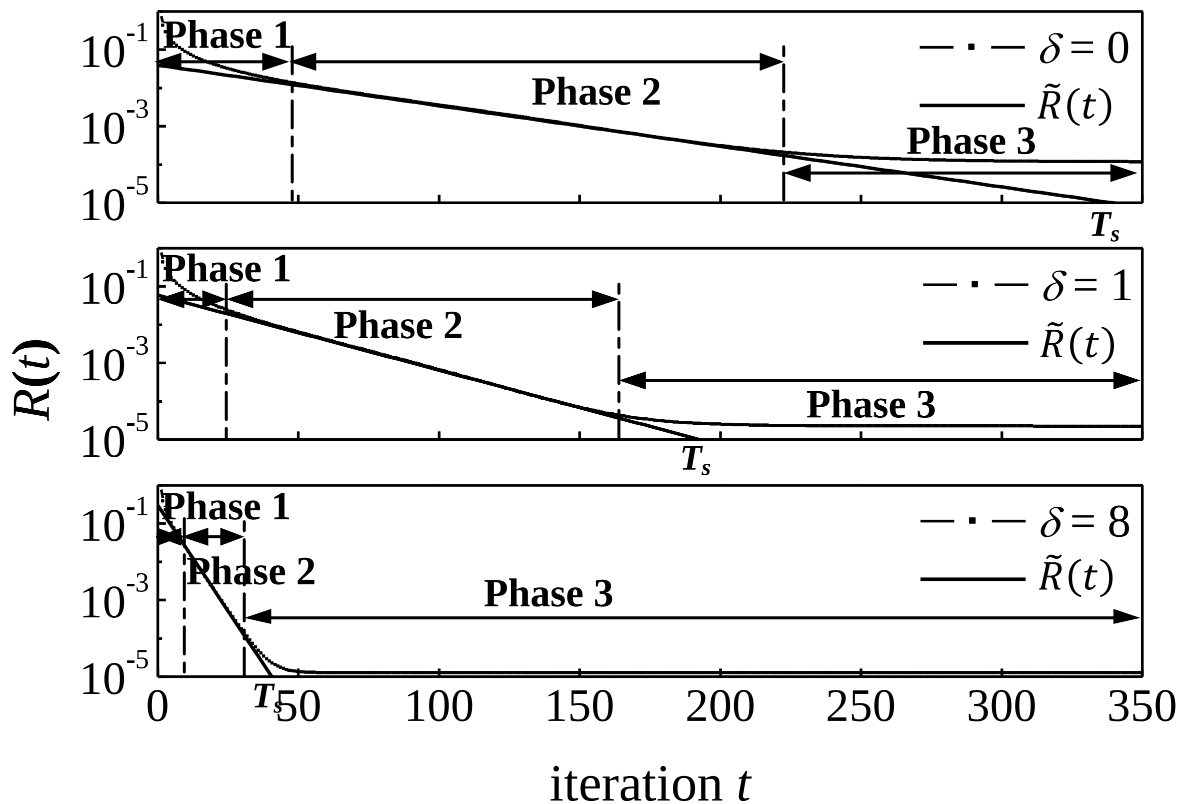
Simulation Results: h

- $|V| = 128$ and $\Delta = 32$



Impact of δ

- Redundant colors speed up the elimination process via don't care edges



Stopping Rule

To achieve a given remaining variable density ϵ, the parallel complex coloring with extra colors δ of a bipartite graph should halt after

$$\frac{\Delta - 1 + \delta}{\Delta - 1 + \delta(1 + a\log(|V| + b) + c)}[a\log(|V| + b) + c]$$

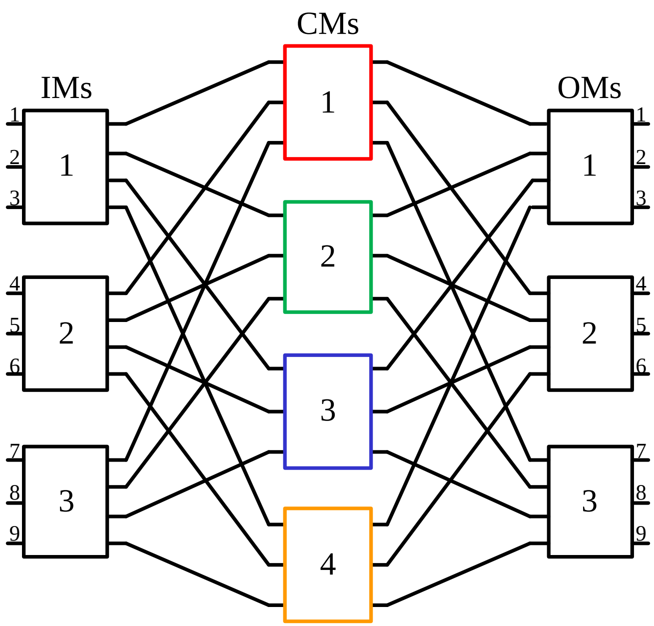
iterations, where a, b, and c are application-specific parameters.

Outline

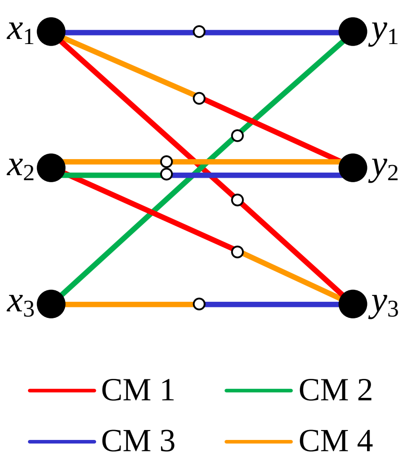
- Introduction and Overview
- Route Assignment and Complex Coloring
- Parallel Complex Coloring with Redundant Colors
- Parallel Routing Algorithm
  - Parallel Routing Assignment Algorithm
  - Performance Evaluation
- Conclusion

Initialization

- Random color assignment.

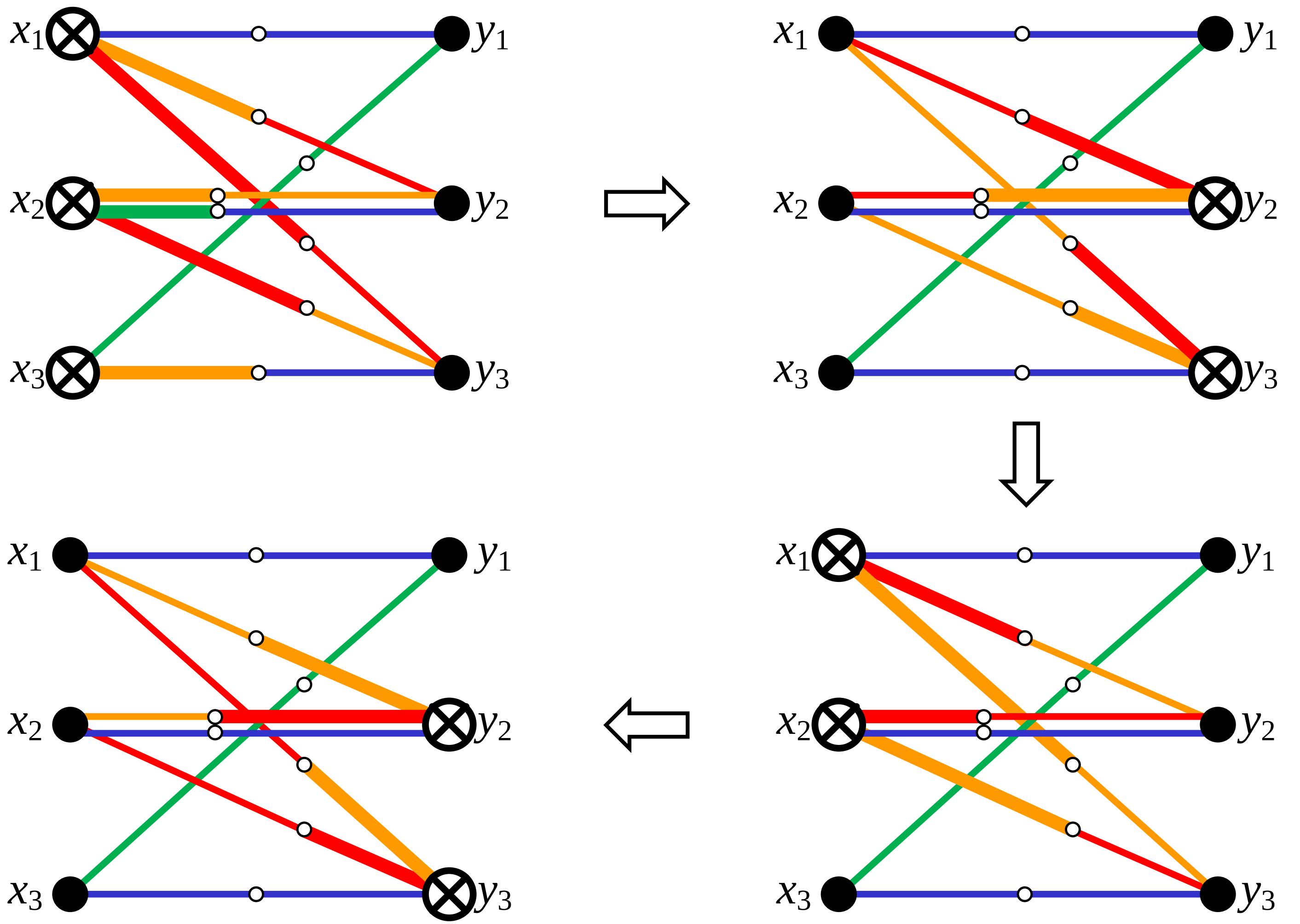


$$\pi = \begin{pmatrix} 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 \\ 5 & 2 & 7 & 8 & 6 & 4 & 9 & 3 & x \end{pmatrix}$$



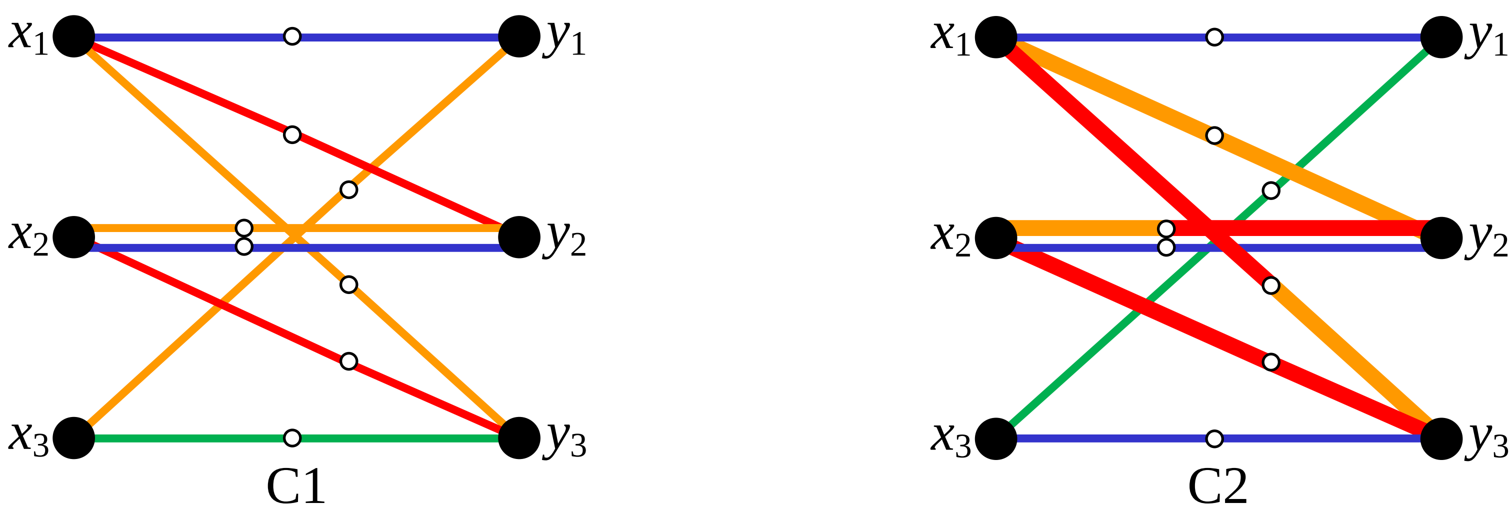
Parallel Complex Coloring

- Perform color exchanges on vertices in X and Y alternatively.



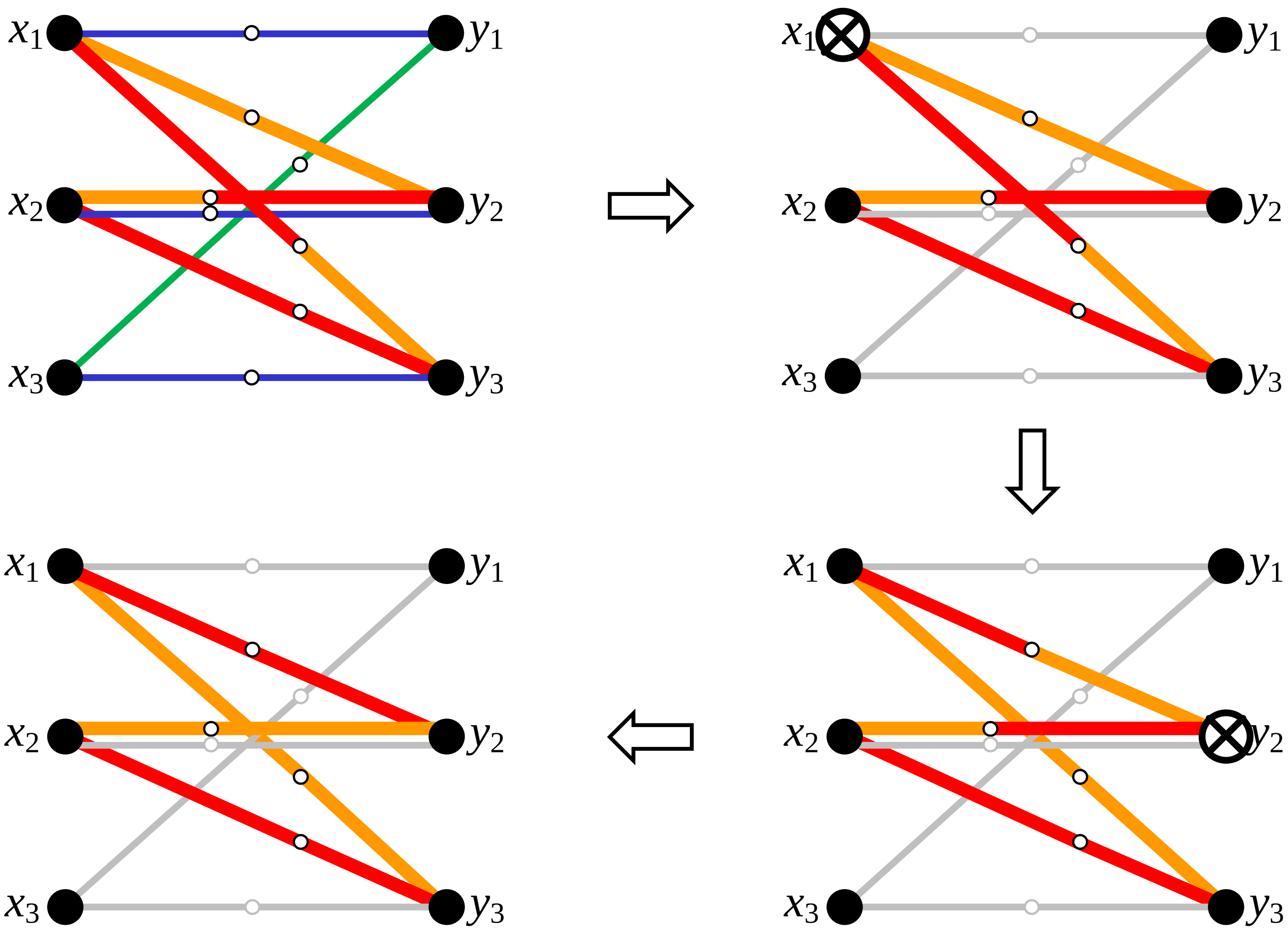
Stopping Condition

- C1: All variables have been eliminated.
- C2: The number of iterations reaches the stopping time.
  - The remaining variables are eliminated by sequential complex coloring.



Sequential Complex Coloring

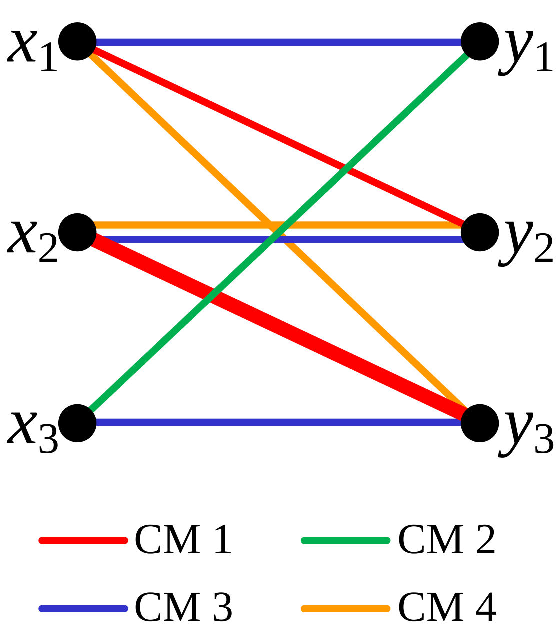
- An example.

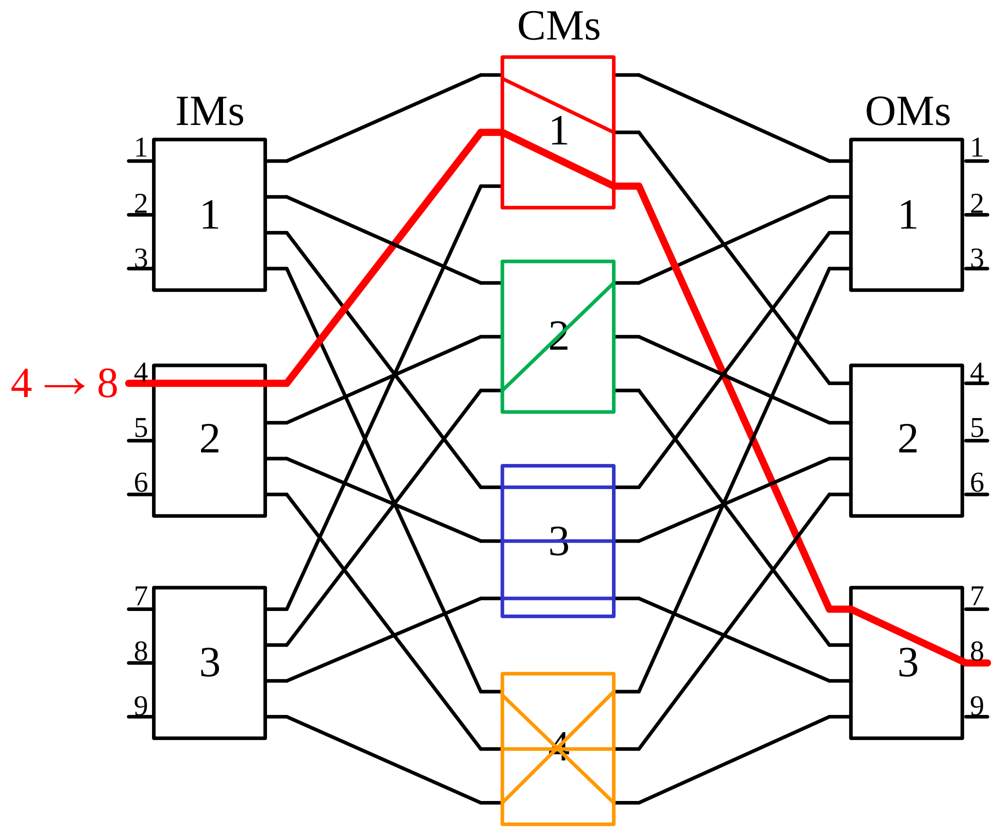


Coloring to Route Assignment

- Edges of the same color constitute a matching that forms a connection pattern of the corresponding central module.

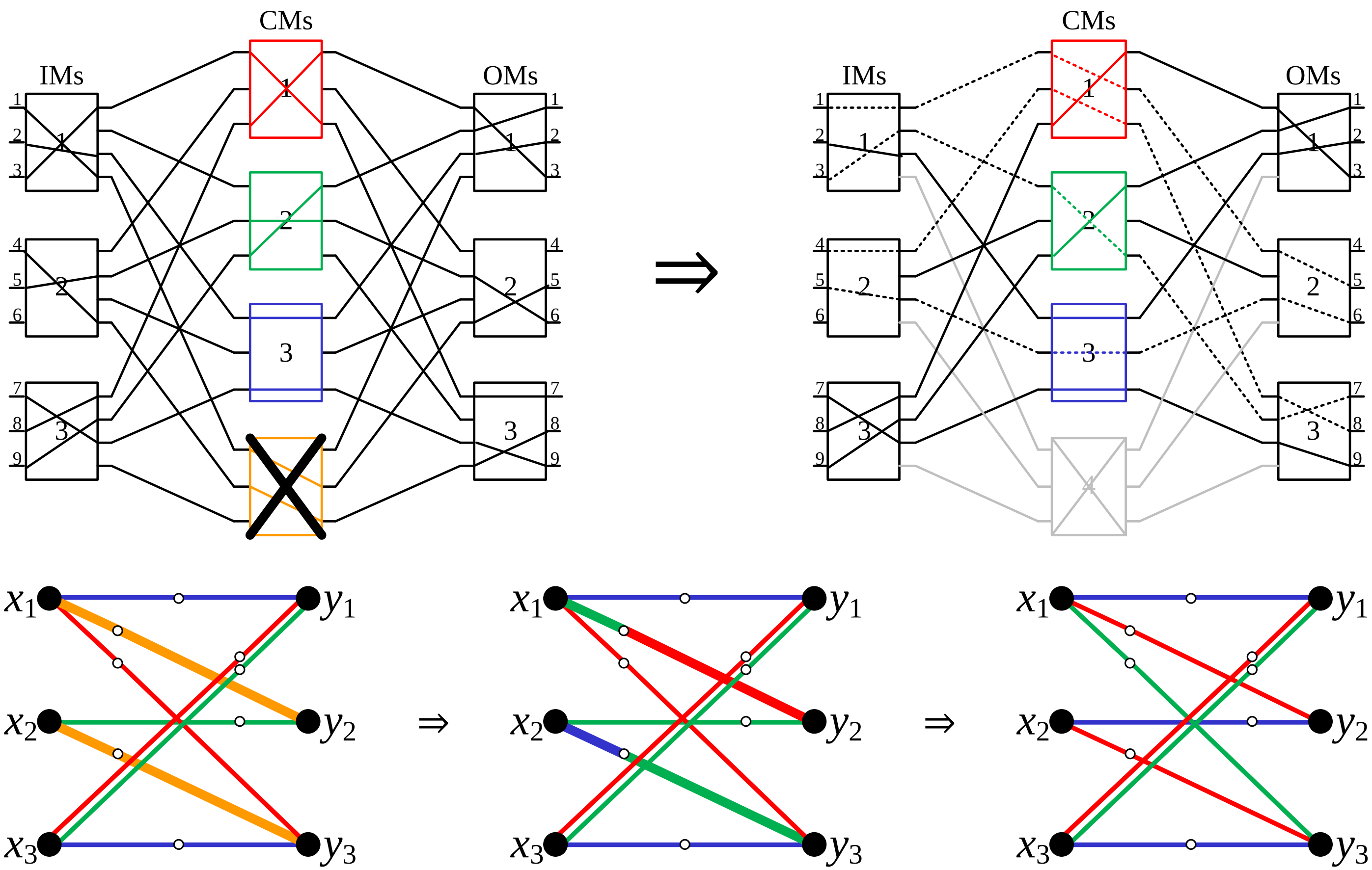
$$\pi = \begin{pmatrix} 1 & 2 & 3 & \boxed{4} & 5 & 6 & 7 & 8 & 9 \\ 5 & 2 & 7 & \boxed{8} & 6 & 4 & 9 & 3 & x \end{pmatrix}$$





Re-routing Upon CM Failure

- Only affected connections are re-established

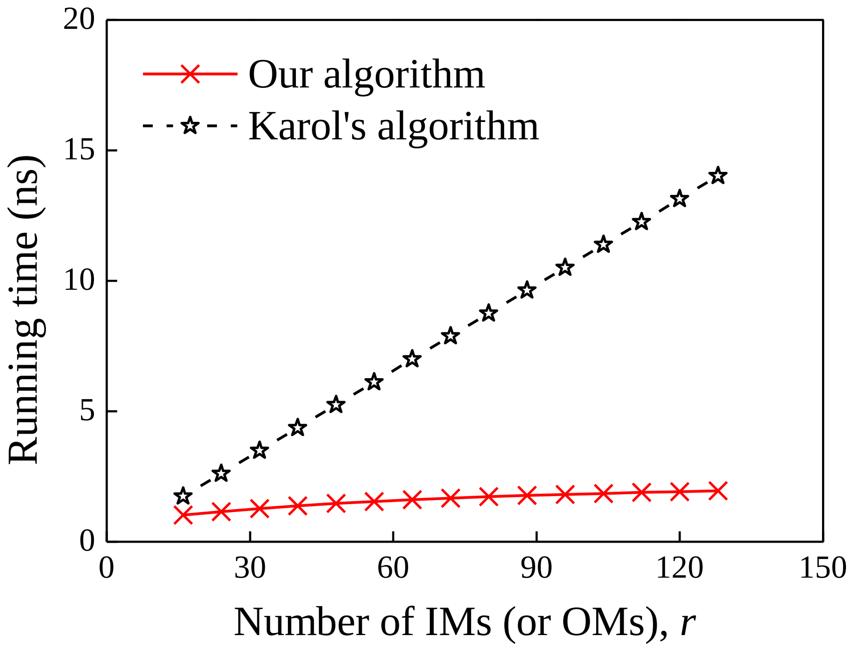


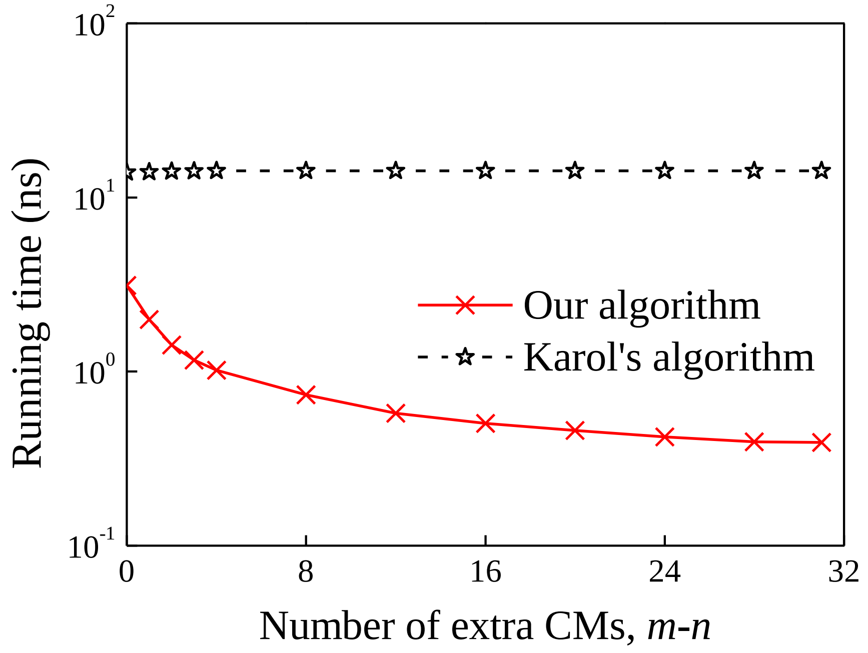
Outline

- Introduction and Overview
- Route Assignment and Complex Coloring
- Parallel Complex Coloring with Redundant Colors
- Parallel Routing Algorithm
  - Parallel Routing Assignment Algorithm
  - Performance Evaluation
- Conclusion

Complexity

- Running time: $O\left(\frac{n(m-1)}{m-1+(m-n)\log r}\log r\right)$
- Extra CMs reduce our running time

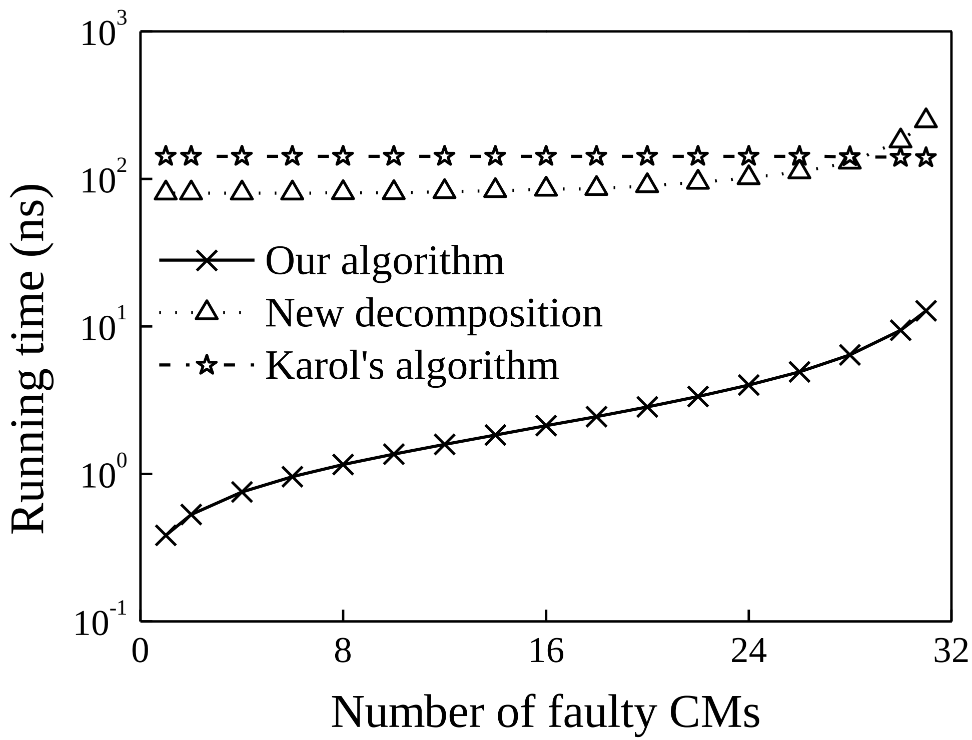




[1] M. J. Karol, and I. Chih-Lin, IEEE/ACM Trans. Commun.

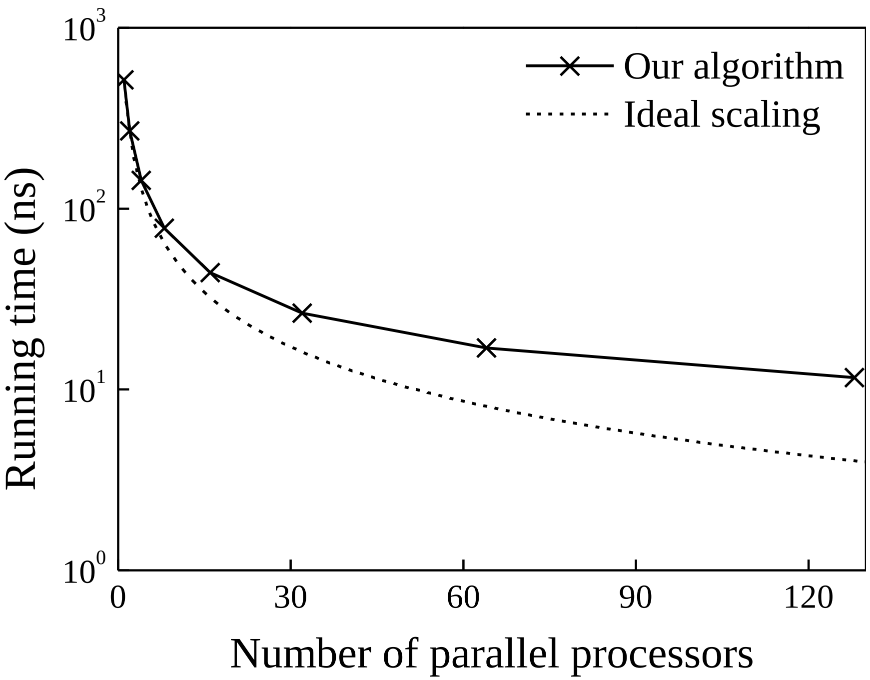
Recovery Complexity

- The rearrangeablility of complex coloring provides a quick recovery from switch failures
  - $C(63,32,128)$

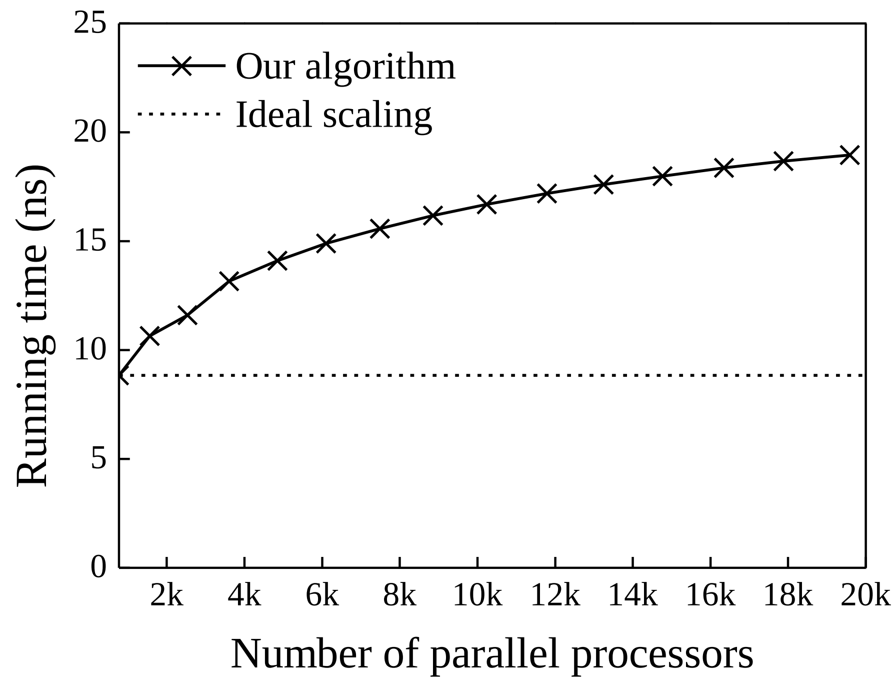


Scalability of Parallelism

- Scalability with multiple parallel processors
  - Strong scaling: refers to the running time of a parallel algorithm versus the number of processors u for a fixed problem size.
  - Weak scaling: refers to the running time of a parallel algorithm versus the number of processors u for a constant amount of work per processor.



(a) Strong scaling



(b) Weak scaling

Outline

- Introduction and Overview
- Route Assignment and Complex Coloring
- Parallel Complex Coloring with Redundant Colors
- Parallel Routing Algorithm
- Conclusion

Conclusion

- Our algorithm can always obtain an optimal route assignment and have 100% bandwidth utilization .
- The time complexity: $O\left(\frac{\sqrt{N}(m-1)}{m-1+(m-\sqrt{N})\log N}\log N\right)$ (m: #of central modules)
  - The minimum order of complexity is $O(\log N)$ with the constant switching module size.
  - The maximum order of complexity is $O\left(\sqrt{N}\log N\right)$ with the switching module size $\Delta = \sqrt{N}$.

[1] T. Lee, Y. Wan, and H. Guan, Randomized Δ-edge colouring via exchanges of complex colours, Int. J. Comput. Math. 90 (2013), 228–245.