Consensus

Distributed Systems

Ali Kamandi, PH.D.

School of Engineering Science

College of Engineering

University of Tehran

kamandi@ut.ac.ir

2024

Process failure models (Benign)

- **Fail-stop**

در این مدل، از یک لحظه به بعد پروسه‌ای که به درستی کار می‌کرده از کار می افتد. سایر پروسه‌ها می‌فهمند که این پروسه دچار اشکال شده است. این مدل یک مدل انتزاعی است و نحوه فهم دیگران از خطا می‌تواند کاملا متنوع باشد.

- **Crash:**

در این مدل یک پروسه که درست کار می‌کرده در یک لحظه از کار می‌افتد. نودهای دیگر از آن مطلع نمی‌شوند.

- **Receive omission:**

در این مدل یک پروسه درست، به نحوی دچار خطا می‌شود که فقط برخی از پیام‌هایی را که برای آن ارسال شده‌است را دریافت می‌کند.

- **Send omission:**

پروسه فقط بعضی از پیام‌هایی را که باید ارسال کند، واقعا ارسال می‌کند.

- **General omission:**

پروسه دچار یکی یا هر دو خطای فوق می‌شود.

Process failure models (Byzantine)

- **Byzantine or malicious failure, with authentication**

  پروسه ممکن است هر رفتار دلخواهی را از خود نشان دهد. اما چنانچه ادعا کند که پیامی را از نود خاصی دریافت کرده است، این ادعا با مکانیزم authentication یا امضا قابل ارزیابی خواهد بود.

- **Byzantine or malicious failure**

  پروسه ممکن است هر رفتار دلخواهی نشان دهد و بر خلاف مدل ادعای دریافت پیام از یک نود سالم، قبل قابل ارزیابی نخواهد بود.

Stopping failure

- **Agreement:** No two processes decide on **different values**.
- **Validity:** If all processes start with the **same initial value** $v \in V$, then v is the only possible decision value.
- **Termination:** All **nonfaulty** processes eventually decide.

FloodSet algorithm (informal)

Each process maintains **a variable W containing a subset of V**. Initially, process i's variable W contains only i's initial value. For each of f+1 rounds, each process broadcasts W, then adds all the elements of the received sets to W.

After f+1 rounds, process i applies the following decision rule. If W is a singleton set, then i decides on the unique element of W; otherwise, i decides on the default value v_0.

FloodSet algorithm (formal)

$\boldsymbol{states_i}$:

$rounds \in \mathbb{N}, initially\ 0$

$decision \in V \cup \{unknown\}, initially\ unknown$

$W \subseteq V, initially\ the\ singleton\ set\ consisting\ of\ i's\ initial\ value$

$\boldsymbol{msgs_i}$:

$if\ rounds \leq f\ then\ send\ W\ to\ all\ other\ processes$

$\boldsymbol{trans_i}$:

$rounds := rounds + 1$

$let\ X_j\ be\ the\ message\ from\ j, for\ each\ j\ from\ which\ a\ message\ arrives$

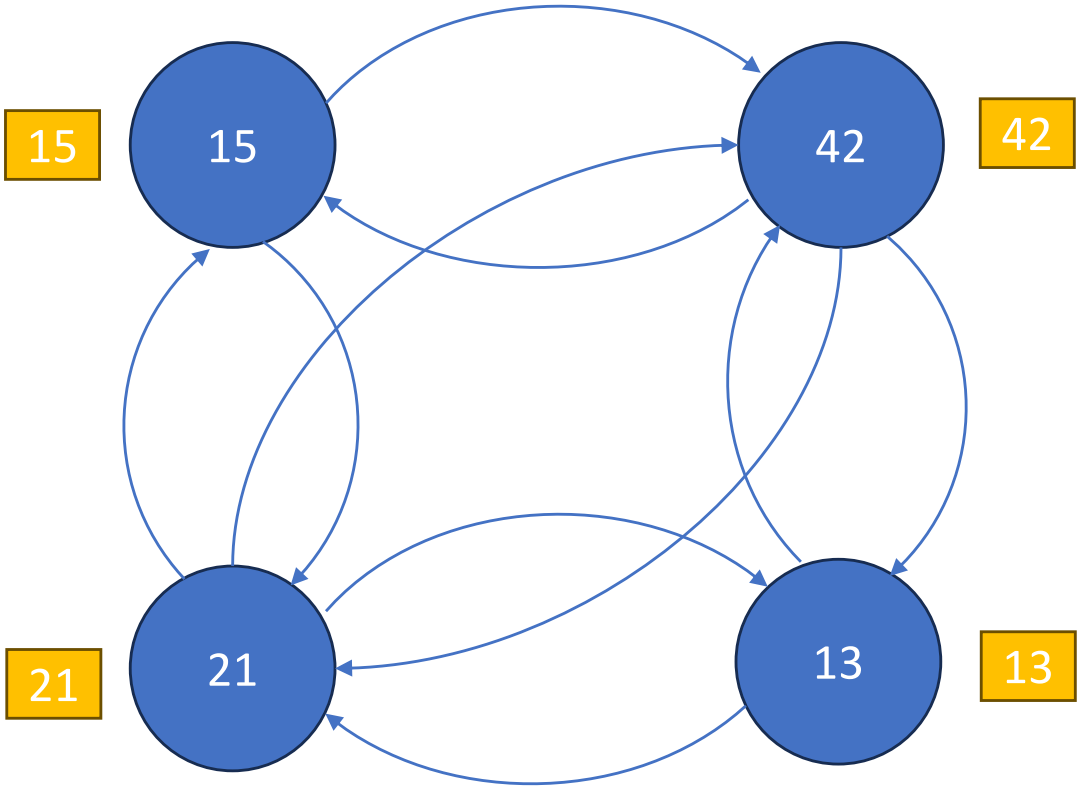
$$W := W \cup \bigcup_j X_j$$

$if\ rounds = f + 1\ then$

$\quad if\ |W| = 1\ then\ decision := v, \qquad where\ W = \{v\}$

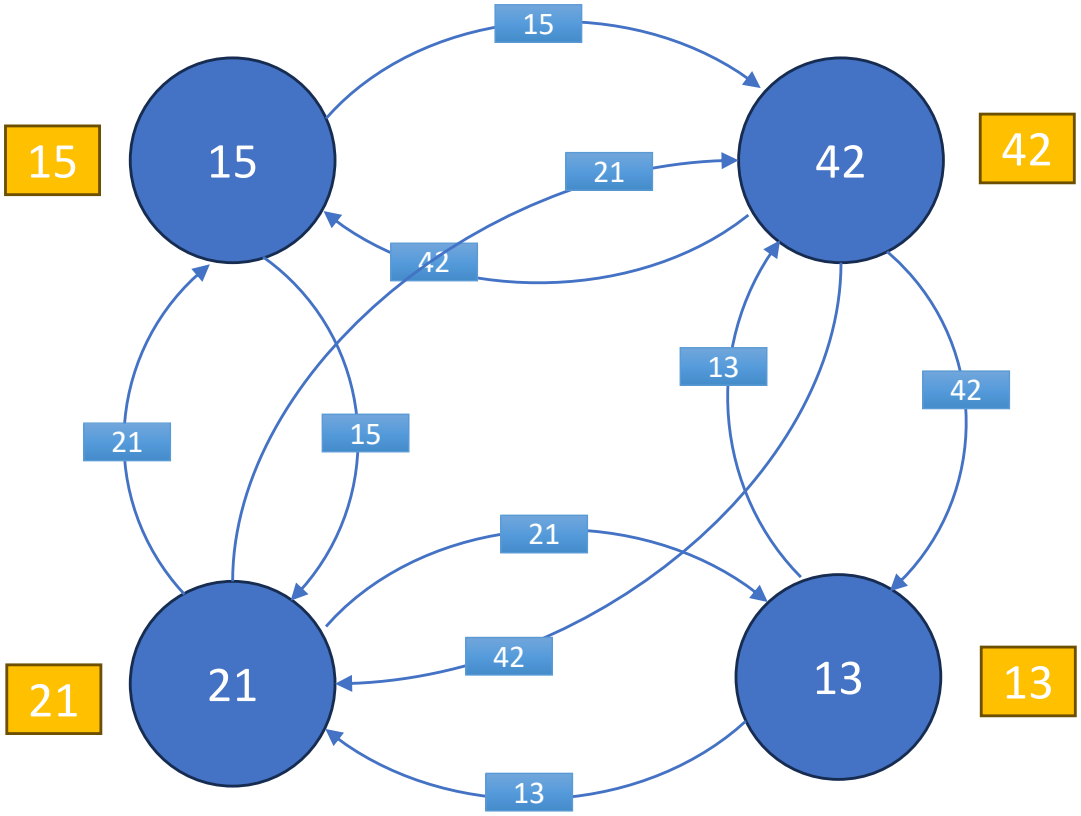
$\quad else\ decision := v_0$

Example



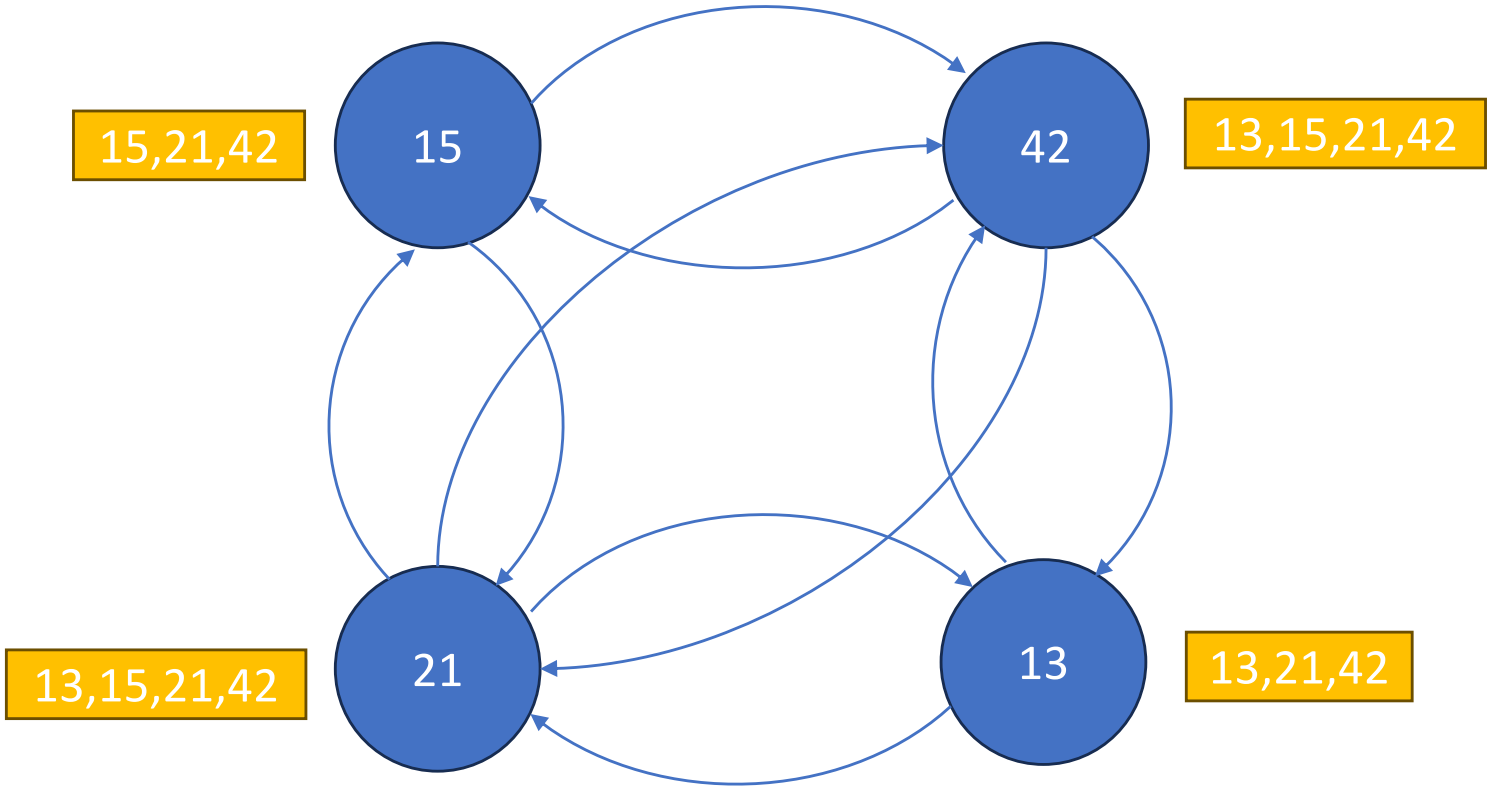
$f = 1, \qquad initial\ state$

Example



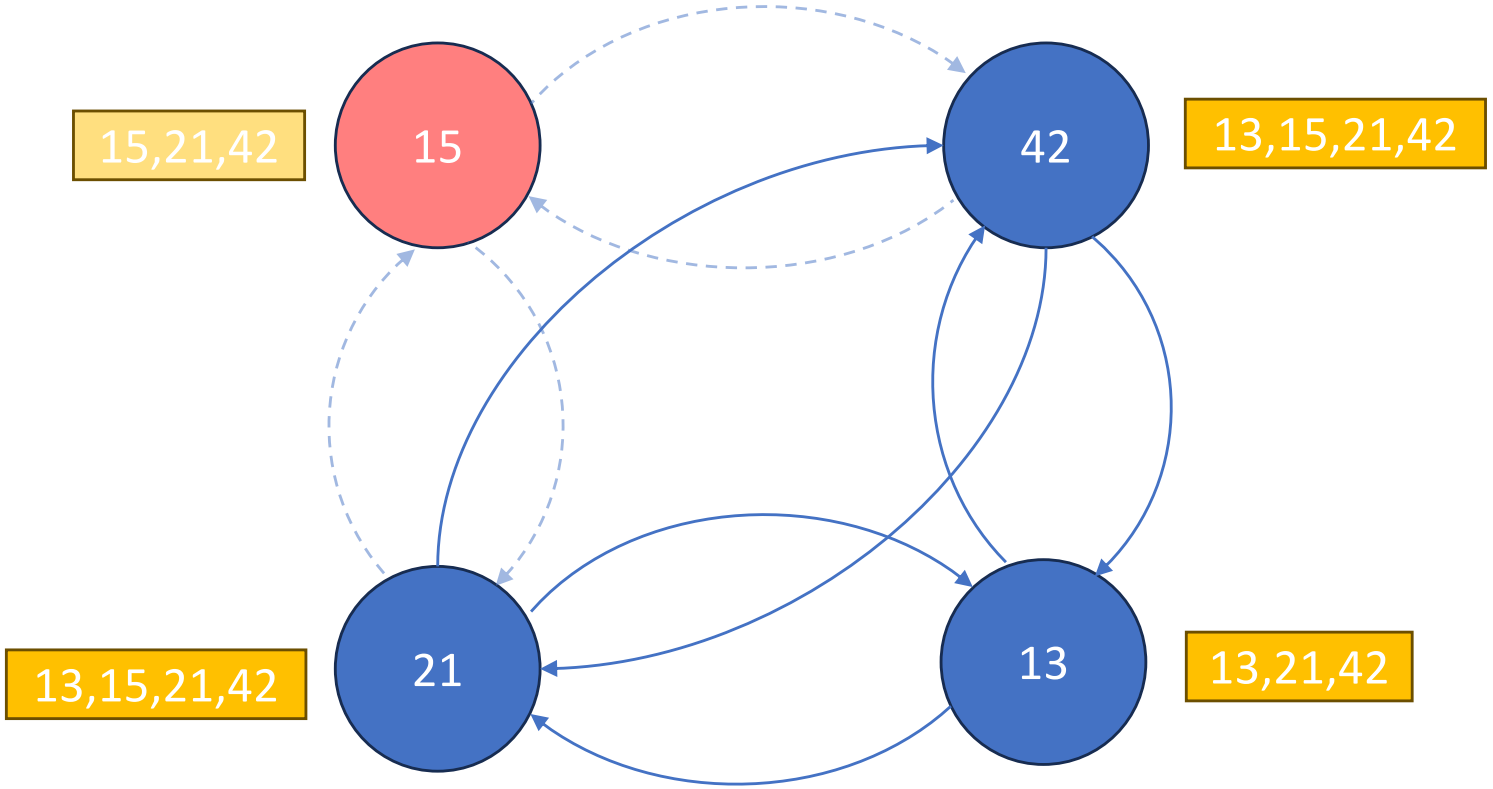
$f = 1, \qquad round\ 1$

Example



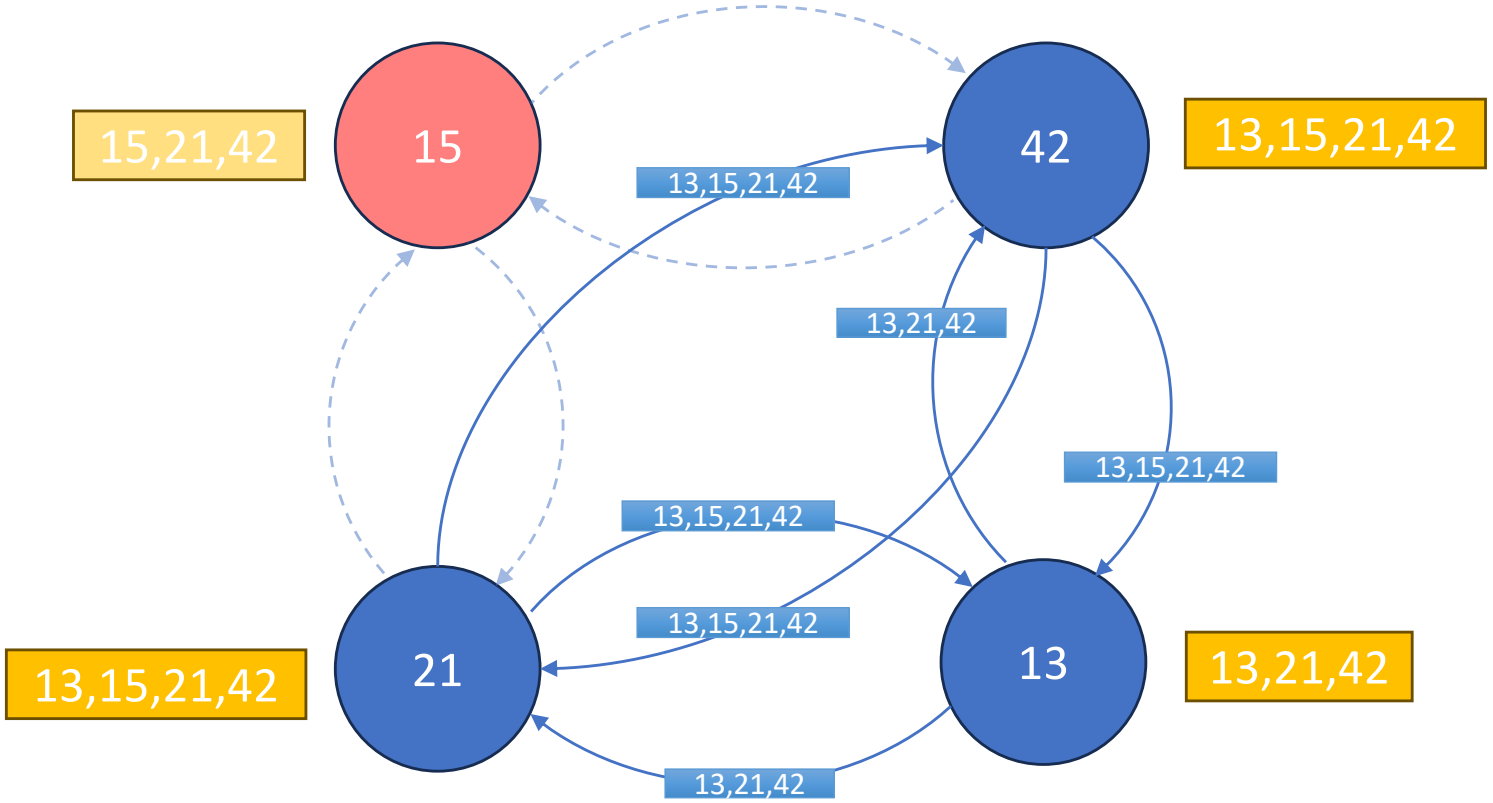
$f = 1, \qquad round\ 1$

Example



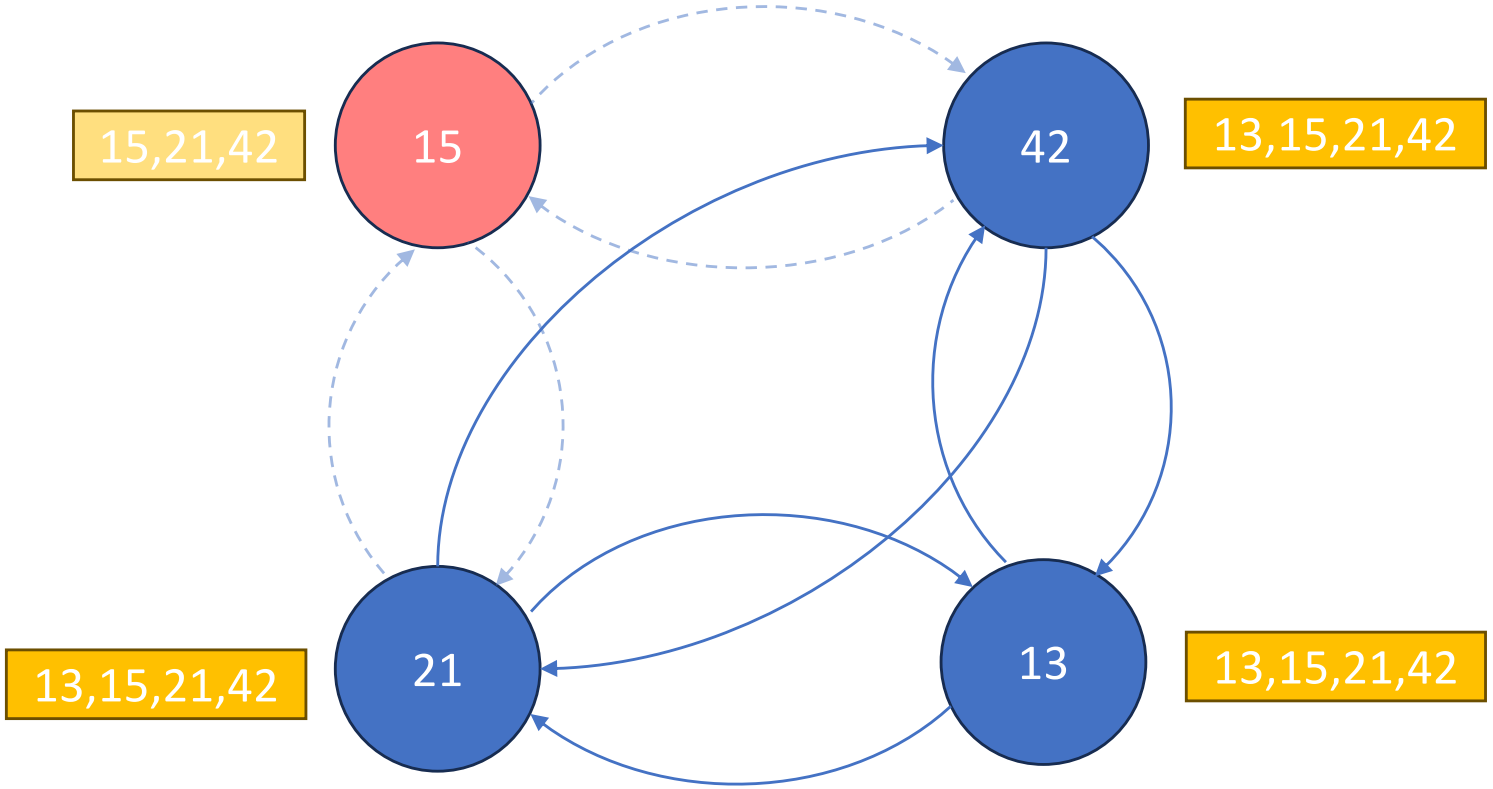
$15\ stop$

Example



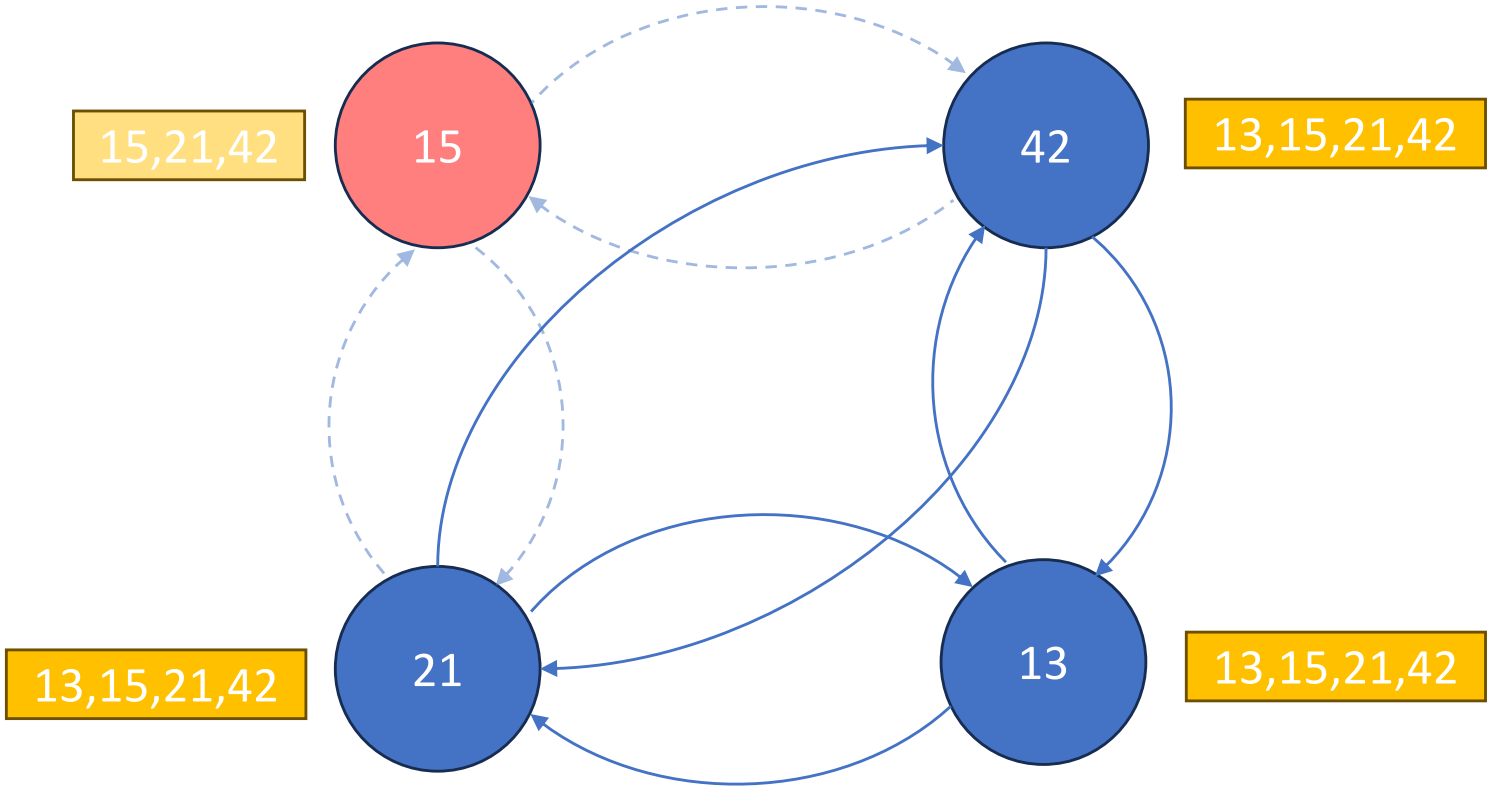
$f = 1, \qquad round\ 2$

Example



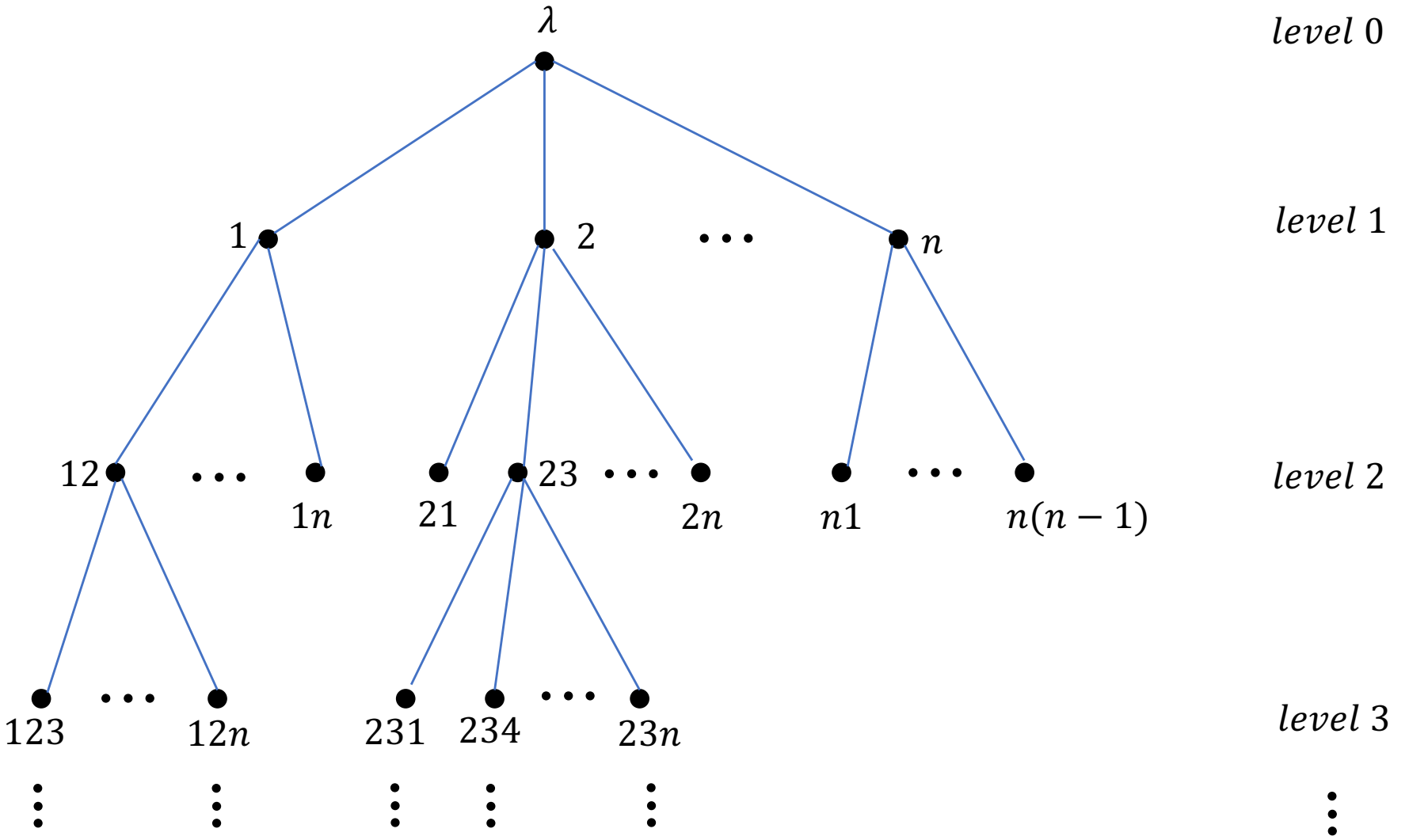
$f = 1, \qquad round\ 2$

Example



$Decision\ is\ v_0$

Exponential Information Gathering Algorithm



EIG Algorithm (informal)

Round 1: process I broadcasts $val(i)$ to all processes, including i itself. Then process i records the incoming information:

1. If a message with value $v \in V$ arrives at i from j, then i sets its $val(j)$ to v.
2. If no message with a value in V arrives at i from j, then i sets $val(j)$ to null.

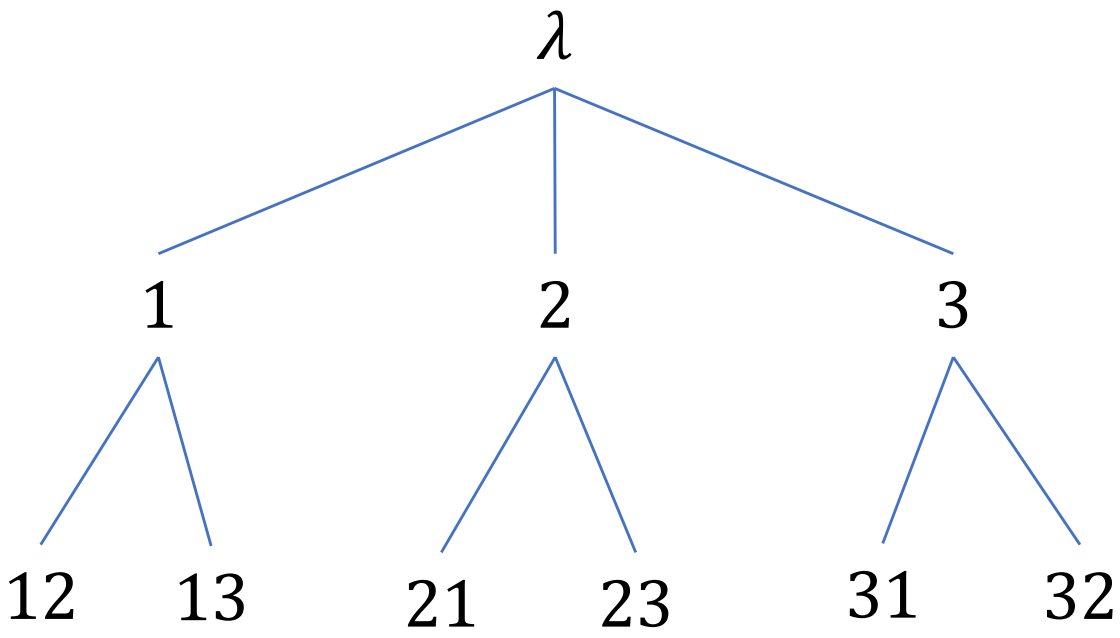
Round k, $2 \leq k \leq f + 1$: Process i broadcasts all pairs (x, v), where x is a level $k - 1$ label in T that does not contain index i, $v \in V$, and $v = val(x)$. Then process i records the incoming information:

1. If x_j is a level k node label in T, where x is a string of process indices and j is a single index, and a message saying that $val(x) = v \in V$ arrives at i from j, then i sets $val(x_j)$ to v.
2. If x_j is a level k node label and no message with a value in V for $val(x)$ arrives at i from j, then i sets $val(x_j)$ to null.

At the end of $f + 1$ rounds, process i applies a decision rule. Namely, let W be the set of non-null values that decorate nodes of i's tree. If W is a singleton set, then i decides on the unique element of W; otherwise, i decides on v_0.

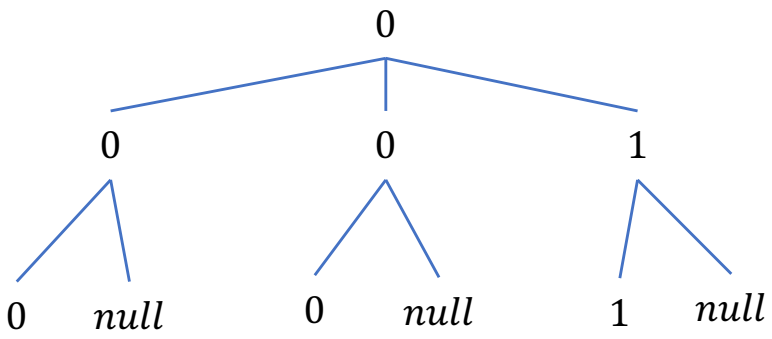
Example

As an example of how the **EIGStop algorithm** executes, consider the case of three processes (n=3), one of which may be fault (f=1). Then the protocol executes for 2 rounds, and the tree has 3 levels. The structure of the EIG tree $T_{3,1}$ here:

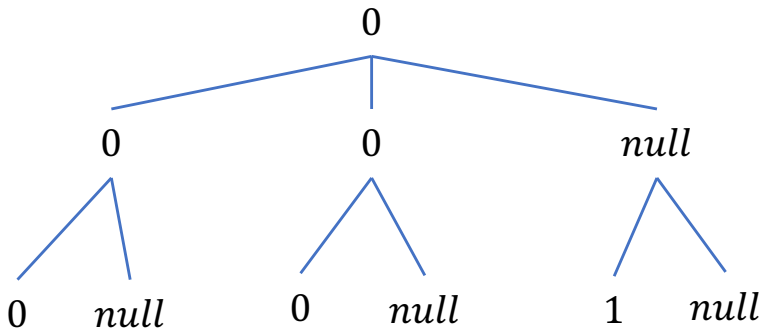


Structure of EIG tree $T_{3,1}$

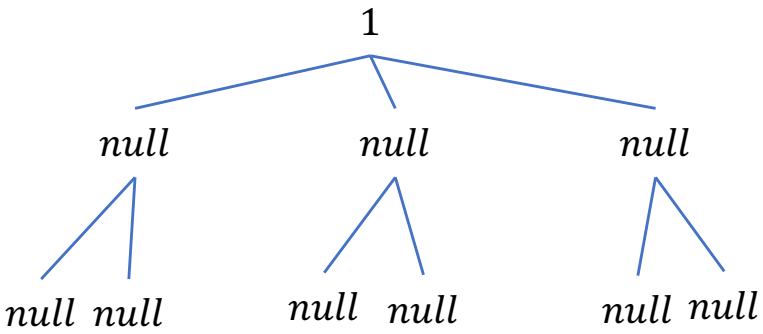
Example



Process 1



Process 2



Process 3

Complexity analysis

The number of rounds is f+1, and the number of messages sent is $\boldsymbol{O((f+1)\,n^2)}$ (This counts each combined message sent by any process to any other at any round as a single message). The number of bits communicated is exponential in the number of failures: $\boldsymbol{O(n^{f+1}b)}$.

Byzantine failure

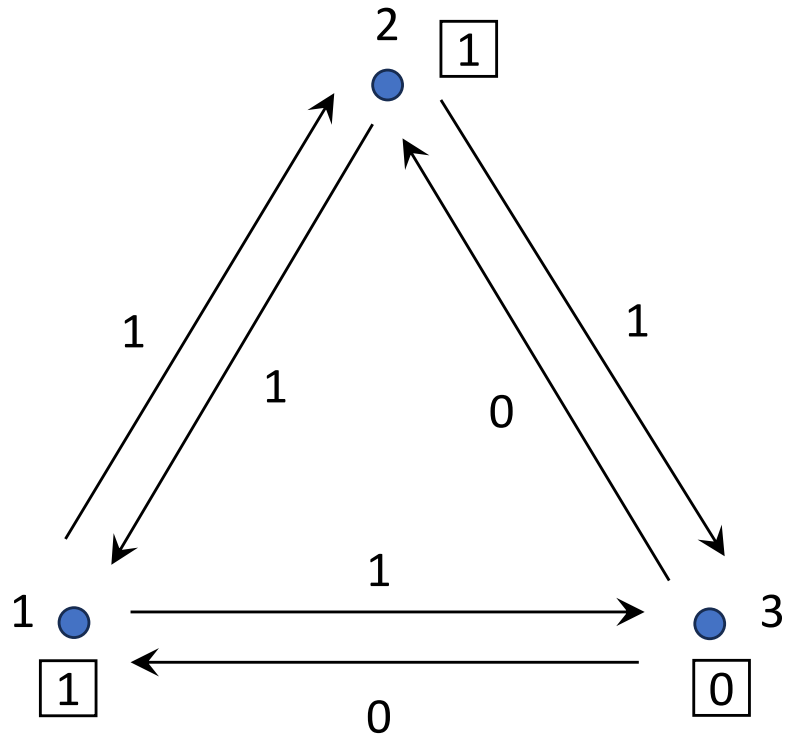
- **Agreement:** No two nonfaulty processes decide on different values.
- **Validity:** If all nonfaulty processes start with the same initial value $v \in V$, then v is the only possible decision value for nonfaulty process.
- **Termination:** The termination condition is the same.

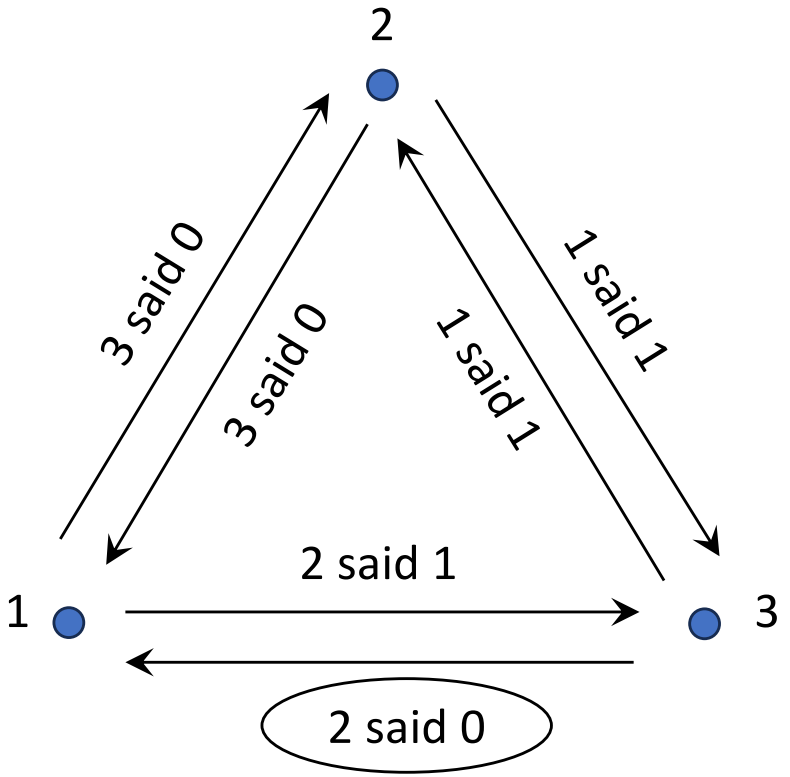
Assumptions

- Complete graph with N nodes
- f is the number of graph's faulty nodes
- Agreed values can be binary or multiple.
- $N > 3f \Rightarrow$ We need $2f + 1$ non-faulty nodes for f faulty nodes.
- *Triple-modular redundancy,*
- A task assign to three process and the majority is considered as the result.

حالت ۱

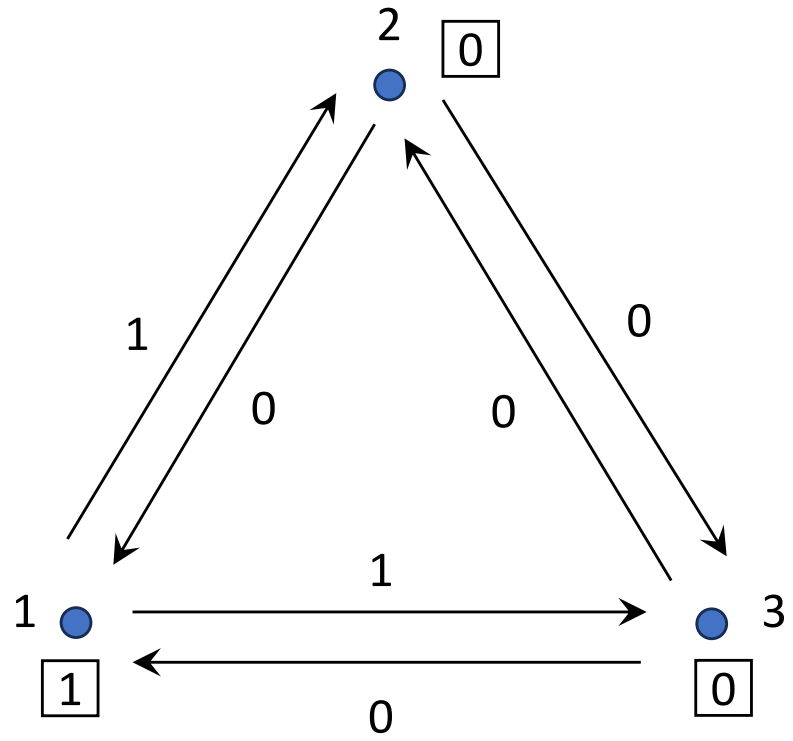
- ۱ و ۲ سالم و ۳ خطادار
- شرط اعتبار: باید روی ۱ توافق شود.

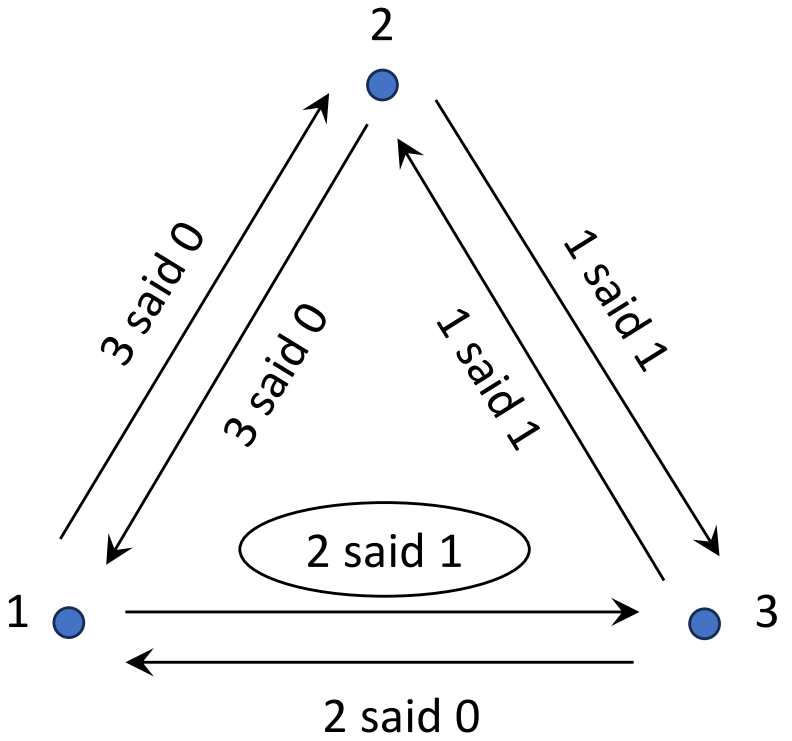




حالت ۲

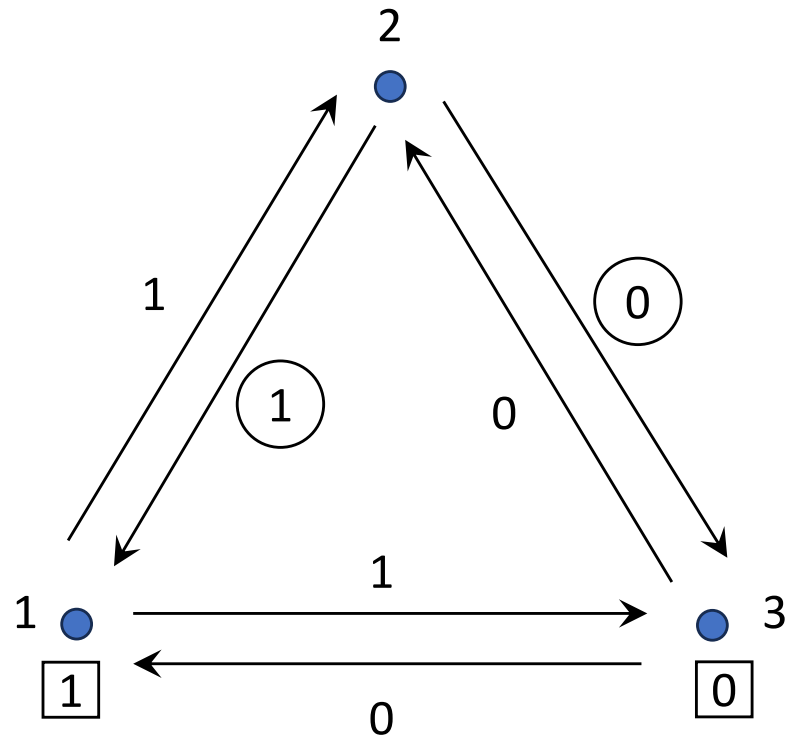
- ۲ و ۳ سالم و ۱ خطادار
- شرط اعتبار: باید روی ۰ توافق شود.

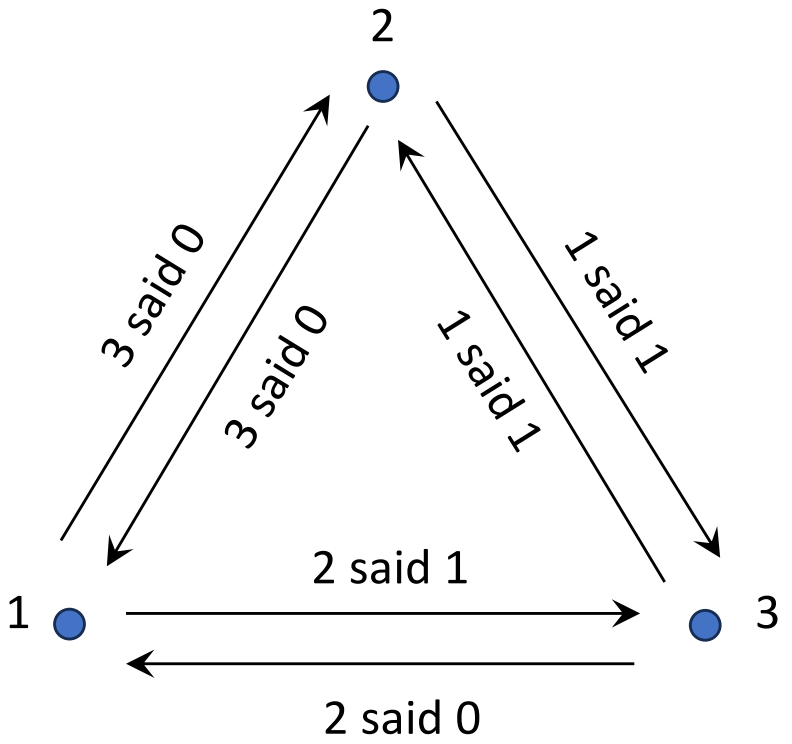




حالت ۳

- ۱ و ۳ سالم و با ۱ و ۰ شروع می‌کنند و ۲ خطادار
- شرط اعتبار: باید روی ۰ توافق شود.





جمع بندی

$\alpha_3 \overset{1}{\sim} \alpha_1$

α_3 and α_1 are indistinguishable to process 1

$\alpha_3 \overset{3}{\sim} \alpha_2$

۱ در حالت ۱ تصمیم ۱ می‌گیرد پس باید در اجرای ۳ هم تصمیم ۱ بگیرد.

۳ در حالت ۲ تصمیم ۰ می‌گیرد پس باید در حالت ۳ هم تصمیم ۰ بگیرد.

در حالت ۳، ۱ تصمیم ۱ و ۳ تصمیم ۰ می‌گیرد که با شرط توافق در تناقض است.

برای سه گره و یک گره خطادار، راه حل وجود ندارد. برای اثبات باید تعداد مراحل بیشتر و حالت‌های کامل‌تری در نظر گرفته شود.

گره ۱ در حالت ۳ می‌داند یکی از گره ها خطا دارد، اما نمی‌داند کدام؟!

EIGByz Algorithm

$N > 3f$

$n = 7$ and $f = 2$

هر نود، مقدار خود را در $f + 1$ مرحله ارسال می‌کند (مشابه EIGStop)، به استثنای اینکه اگر پیامی دریافت کند که معیوب باشد، دور انداخته می‌شود (همانند وقتی که پیامی دریافت نکرده باشد رفتار می‌شود).

پس از مرحله $f+1$ تصمیم گیری صورت می‌پذیرد.

نحوه تصمیم‌گیری

همه مقادیر null با مقدار پیش فرض v0 جایگزین می‌شوند.

برای تصمیم‌گیری نهایی، هر نود به هر یک از گره‌ها یک مقدار جدید تخصیص می‌دهد.

این کار از برگ ها آغاز می‌شود.

برای برگ‌ها: $newval(x) := val(x)$.

برای گره‌های غیر برگ: مقدار جدید بر اساس اکثریت مطلق مقادیر فرزندان تعیین می‌شود. چنانچه اکثریت وجود نداشت، مقدار پیش فرض در نظر گرفته می‌شود.

تصمیم نهایی نود i برابر است با:

$newval(\lambda)$

Complexity analysis

The number of rounds is $f + 1$, and the number of messages sent is $\boldsymbol{O((f+1)n^2)}$ (This counts each combined message sent by any process to any other at any round as a single message). The number of bits communicated is exponential in the number of failures: $\boldsymbol{O(n^{f+1}b)}$.