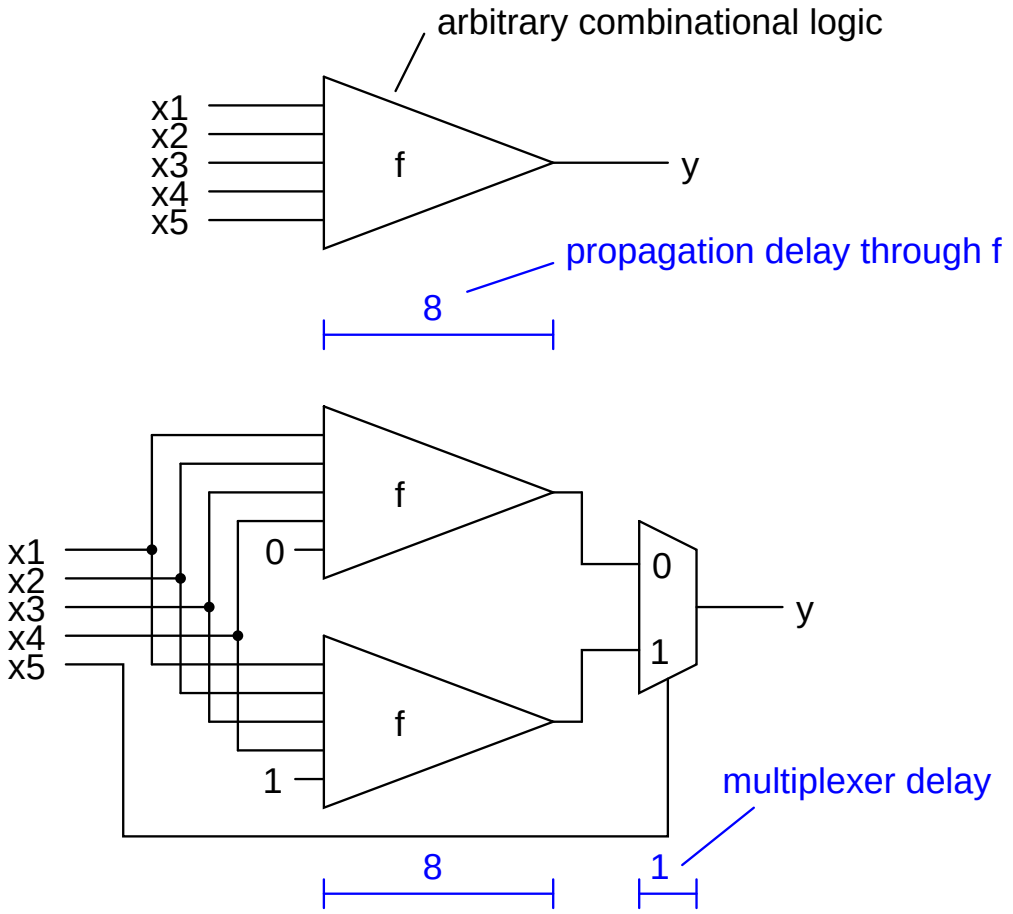
High level optimization by Retiming and Shannon decomposition

Cristian Soviani, Olivier Tardieu, Stephen A. Edwards

Department of Computer Science,
Columbia University, 2005

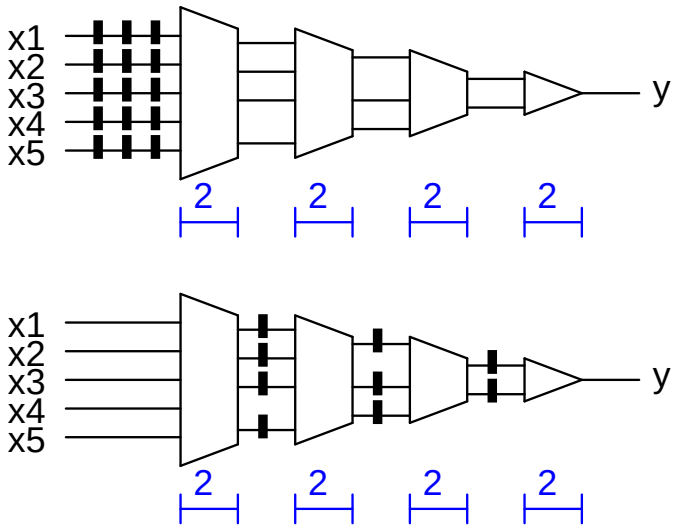
{soviani,tardieu,sedwards}@cs.columbia.edu

Shannon transform - review



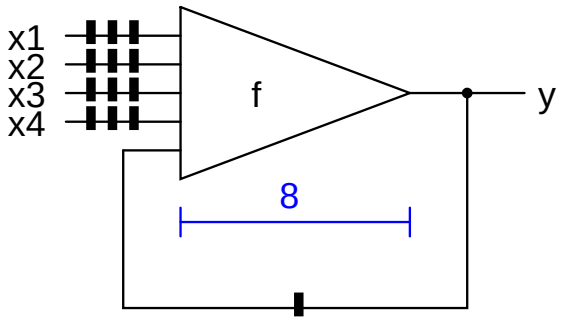
Improves performance if x5 is late compared to x1...x4

Retiming - review



Improves performance by re-distributing the registers

Motivation. Let's speed up this sample

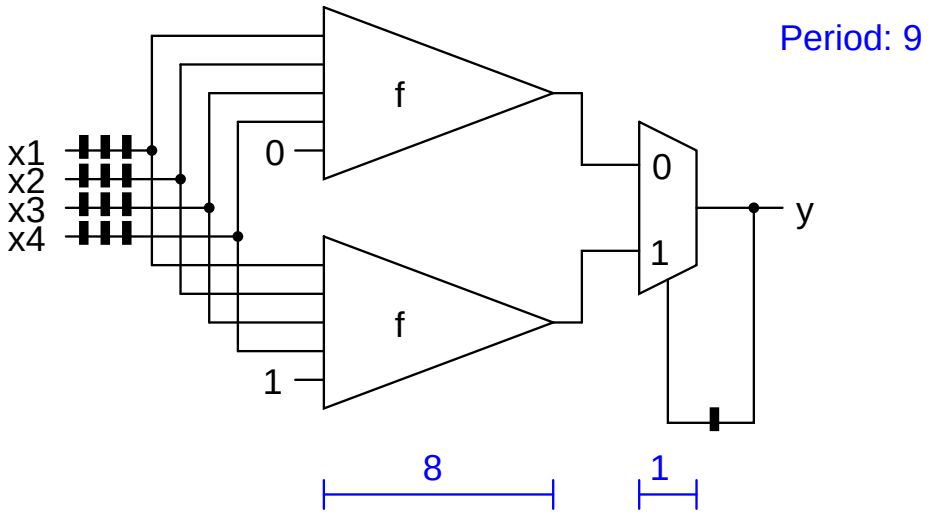


Period: 8

Observations:

- Retiming can not improve performance because of the loop
- Shannon seems useless, as all inputs arrive at the same time (0)

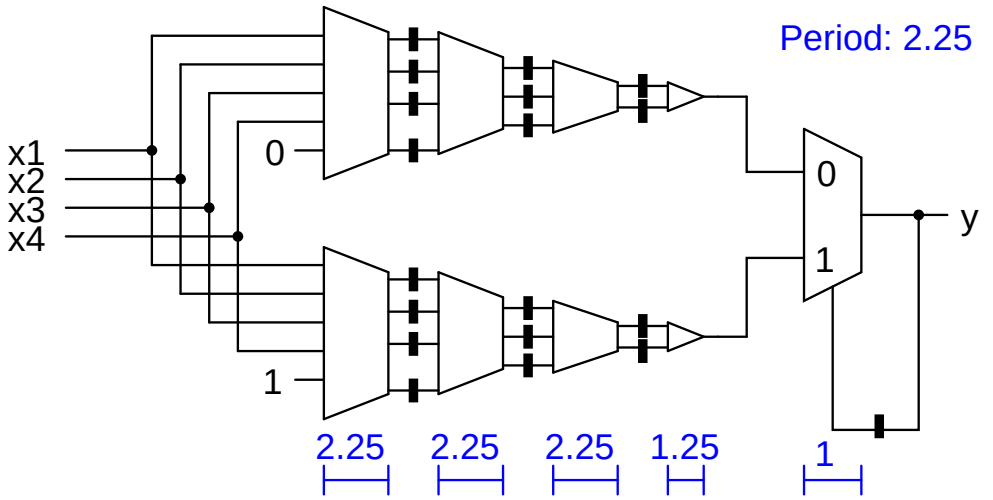
Motivation sample after Shannon transform



Observations:

- the performance is worse
- the loop is much smaller

Motivation sample after Shannon and retiming



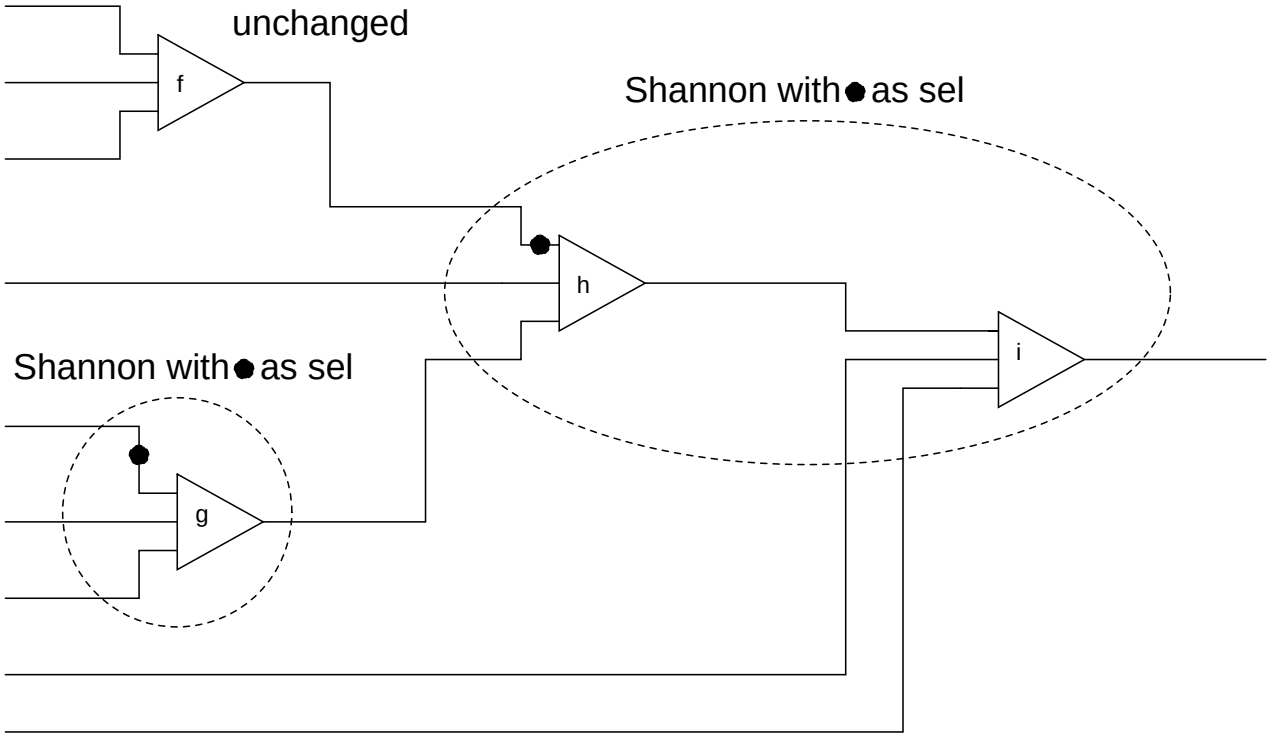
Shannon transform(s) and retiming: a huge design space

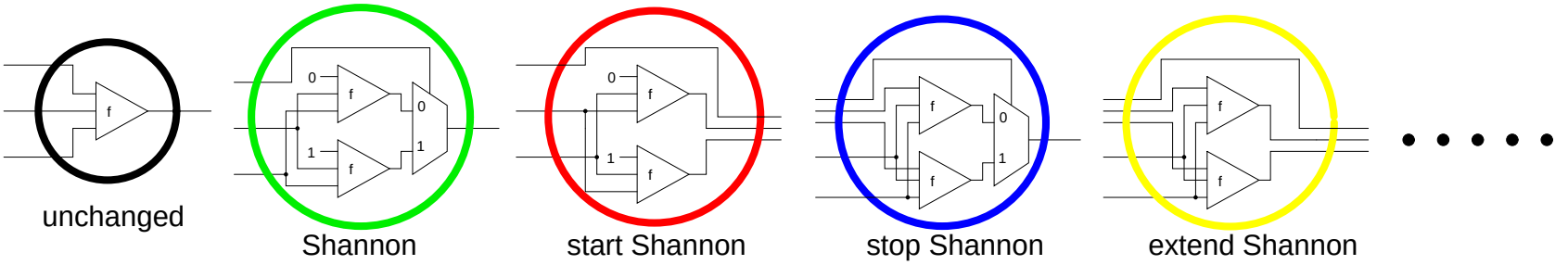
- finding the best combination is not trivial
- we want a systematic way to explore it

Presentation outline. Contributions

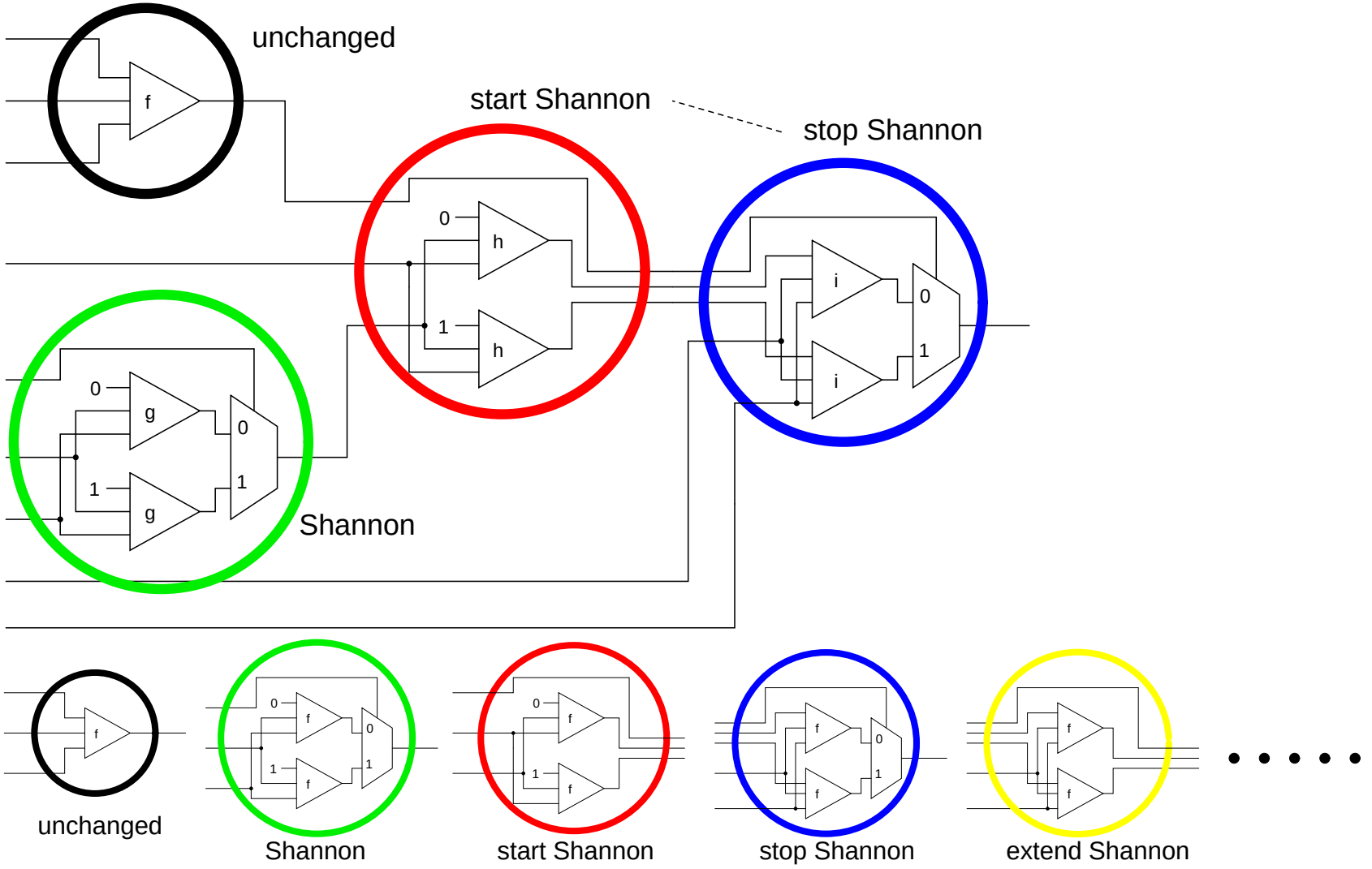
- Shannon transform and retiming — review
- motivation sample
- proposed view of complex combinations of Shannon xforms — “feasible arrival times”
- critical cycles: the fundamental limit of retiming — review
- proposed algorithm: systematic exploration of the Shannon / retiming design space
  - simulation for a small sample
- experimental results and conclusions

Shannon decompositions seen as covering

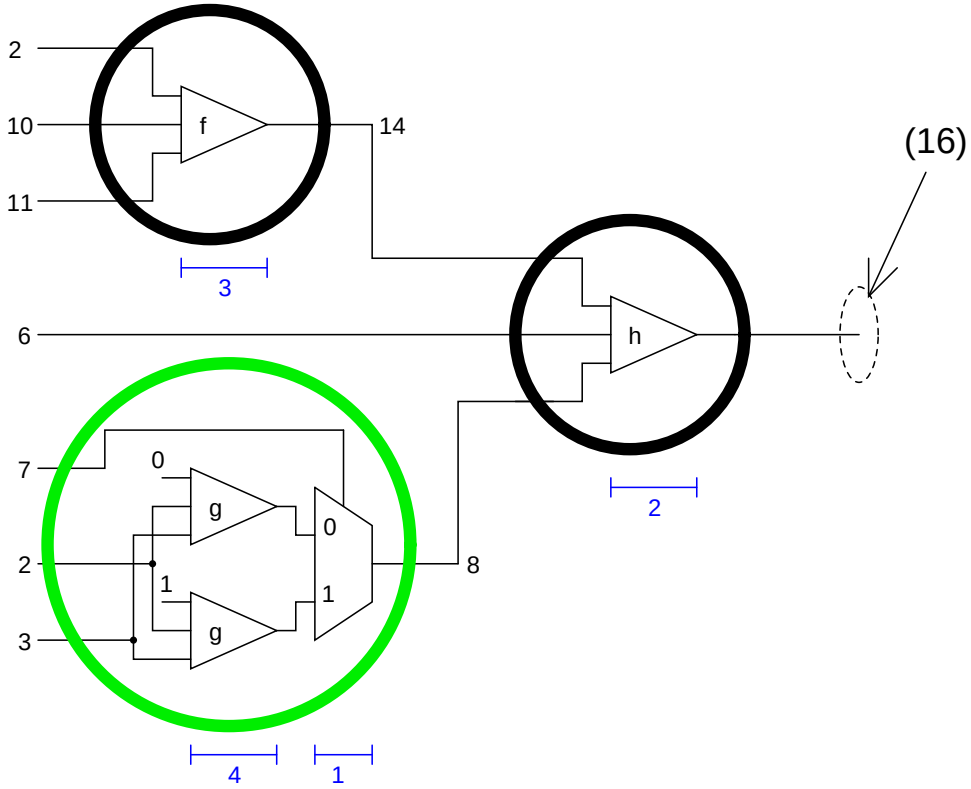


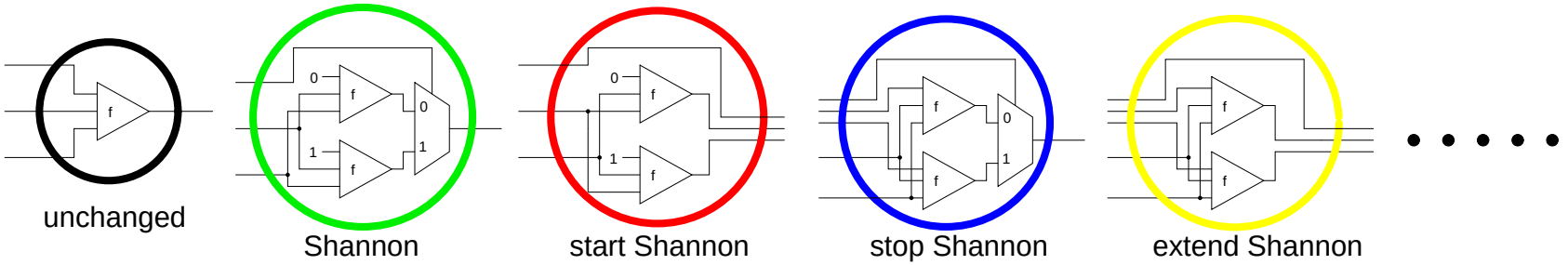


Shannon decompositions seen as covering

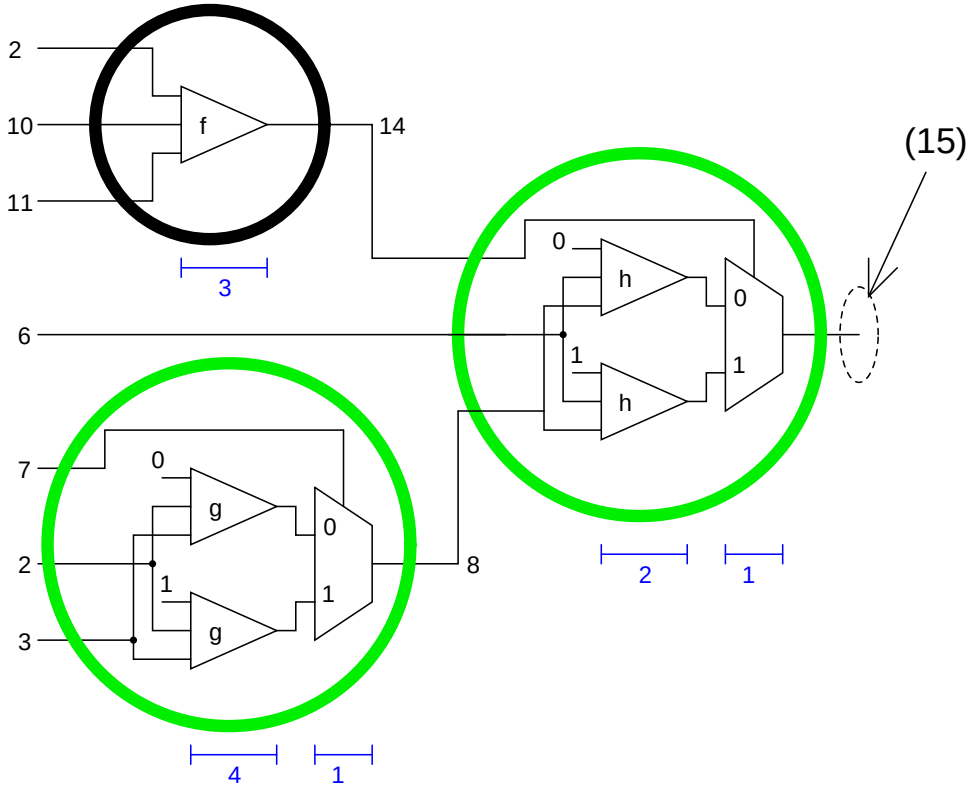


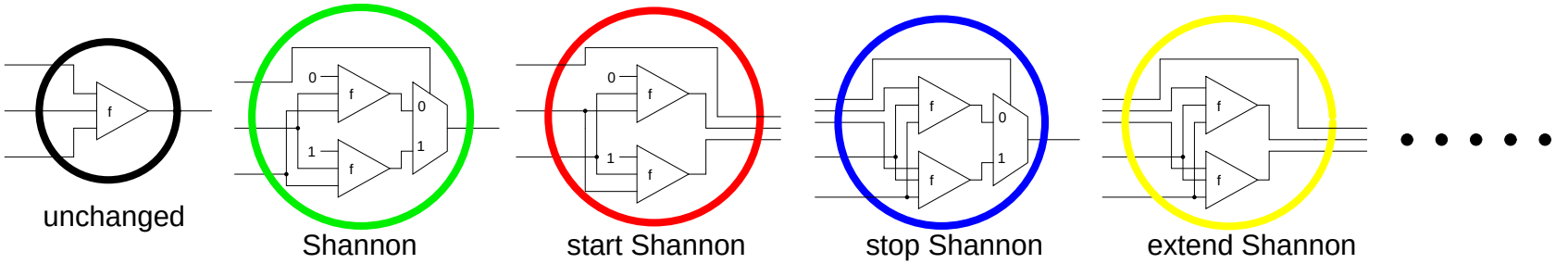
Exploring variants for h: “unchanged”



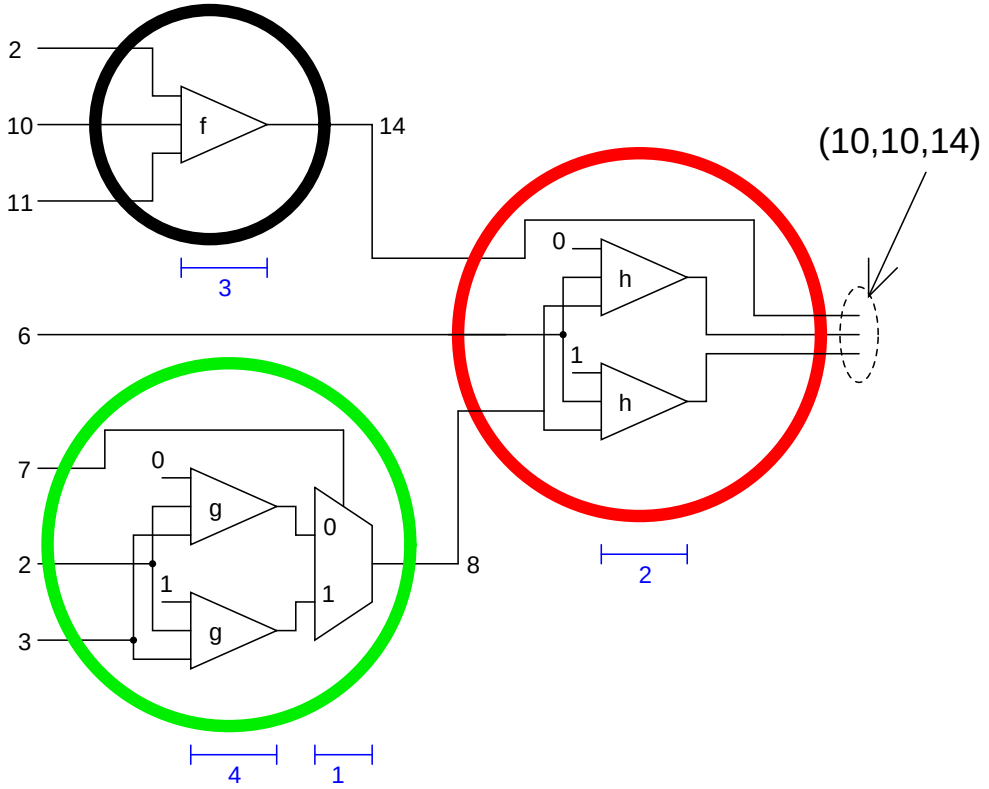


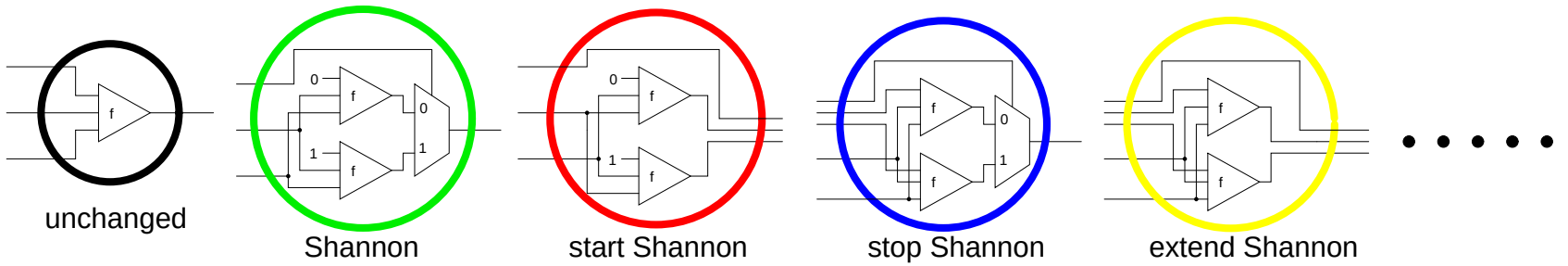
Exploring variants for h: “Shannon”



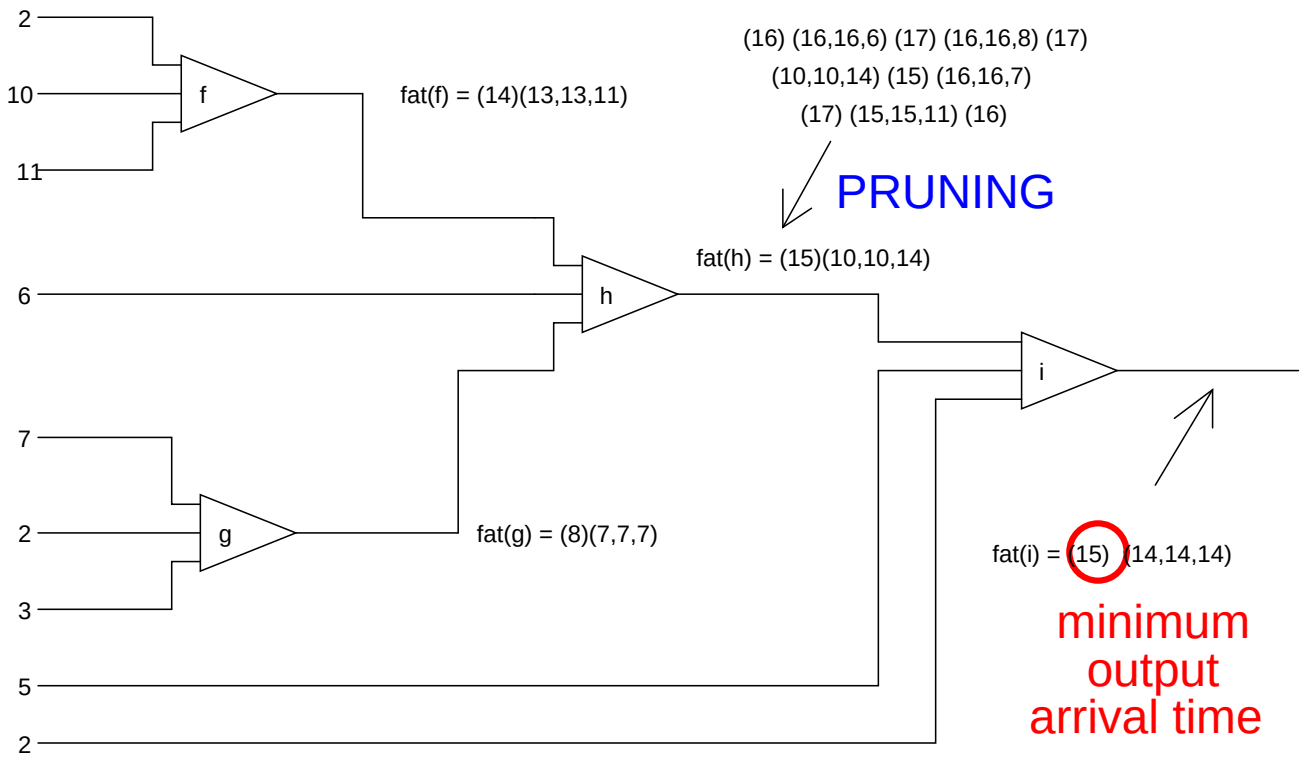


Exploring variants for h: “Start Shannon”



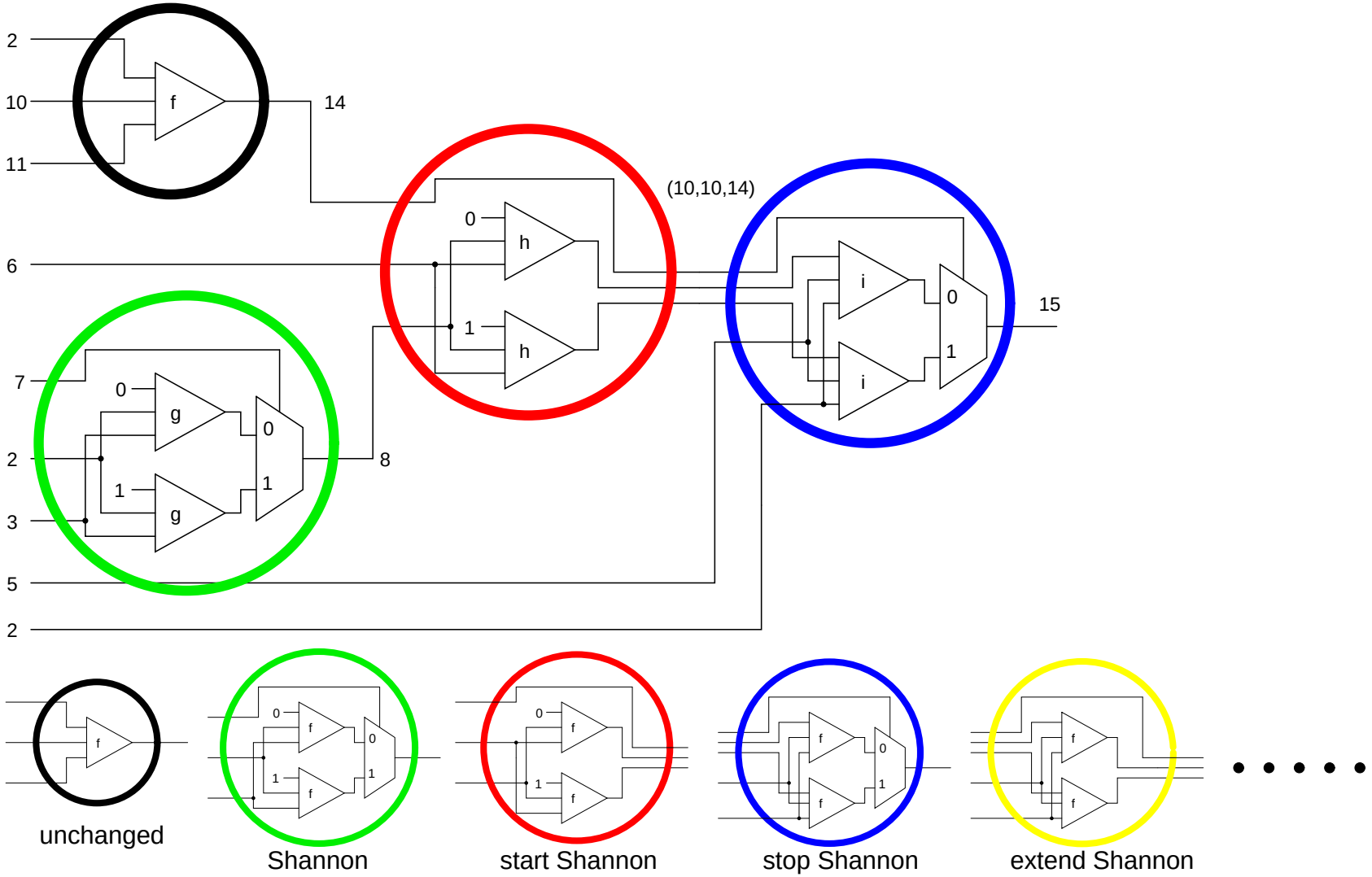


All possible “Feasible arrival times”

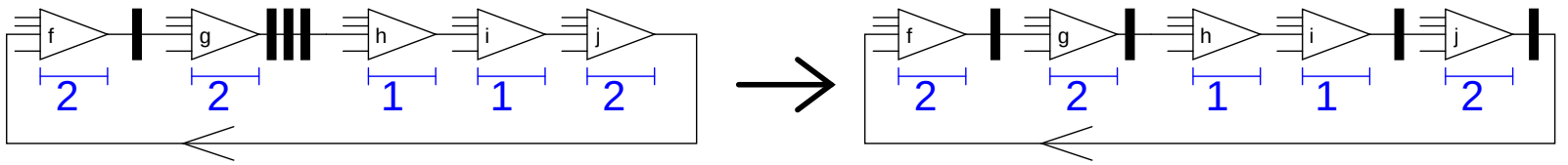


$$\begin{aligned} \mathit{fat}(h) &= \text{combine}(d(h), \{\mathit{fat}(f), \{(6)\}, \mathit{fat}(g)\}) \\ &= \{(15), (10, 10, 14)\} \end{aligned}$$

Solution


2
10
11
f
14
6
0
h
1
h
(10,10,14)
i
0
1
i
15
7
2
3
0
g
1
g
0
1
8
5
2
unchanged
Shannon
start Shannon
stop Shannon
extend Shannon

Retiming limitation for one cycle



$$
\begin{aligned}
c &\geq \frac{\text{delay}_{\text{cyc}}}{\text{regs}_{\text{cyc}}} \\
\text{delay}_{\text{cyc}} &\leq c \cdot \text{regs}_{\text{cyc}} \\
(\text{delay}_{\text{cyc}} - c \cdot \text{regs}_{\text{cyc}}) &\leq 0
\end{aligned}
$$

Assign weight $(-c)$ to registers:

Period c is feasible $\Leftrightarrow$ the cycle has negative weight

Retiming limitation for all the circuit

Period c is feasible if ALL cycles are negative.

Bellman-Ford detects positive cycles in polynomial time

$$\left\{ \begin{array}{l} \text{Period c is feasible} \\ \text{ALL cycles are negative} \\ \text{Bellman-Ford converges to a FIX POINT} \end{array} \right\}$$

Key: fix point equation (holds only if BF converges !!!)

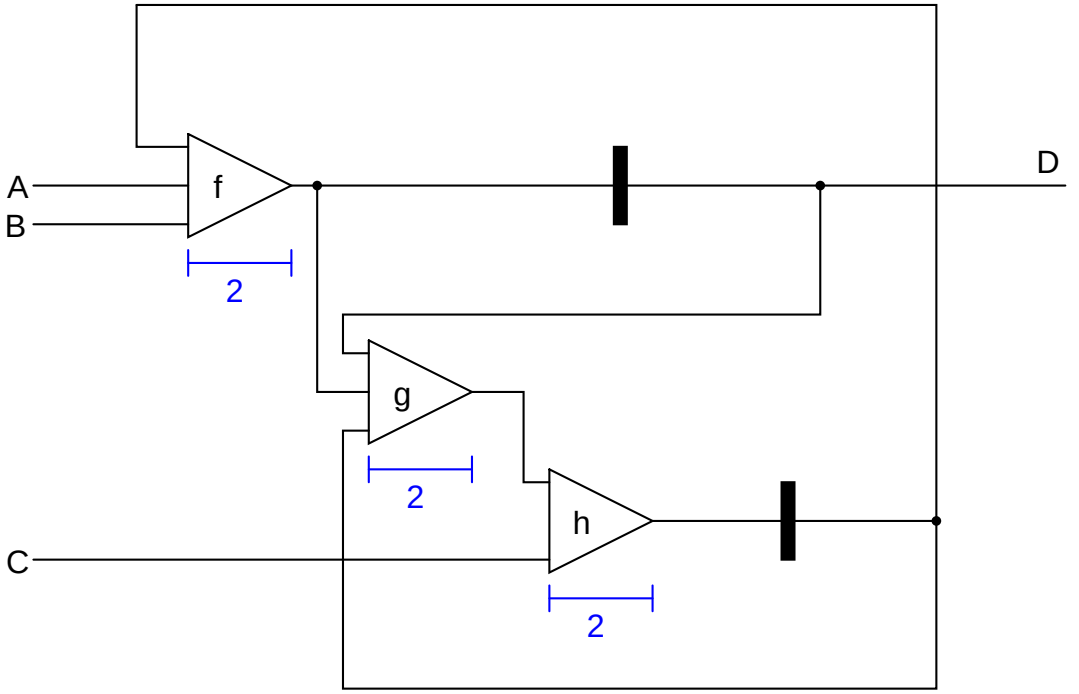
$$\mathit{fat}(n) = \text{combine}(d(n), \{\mathit{fat}(n')\}_{n' \in \text{fanins}(n)})$$

Algorithm outline

```
procedure SeqShannon(S, c)
    (converges, fix_point_fat) = Bellman-Ford (S, c)
    if not converges then
        return NOT_FEASIBLE
    ShannonTransform(S, c, fix_point_fat)
    Retime(S)
    return SUCCESS
```

- we can approximate the best period c by binary search

Sequential sample - original



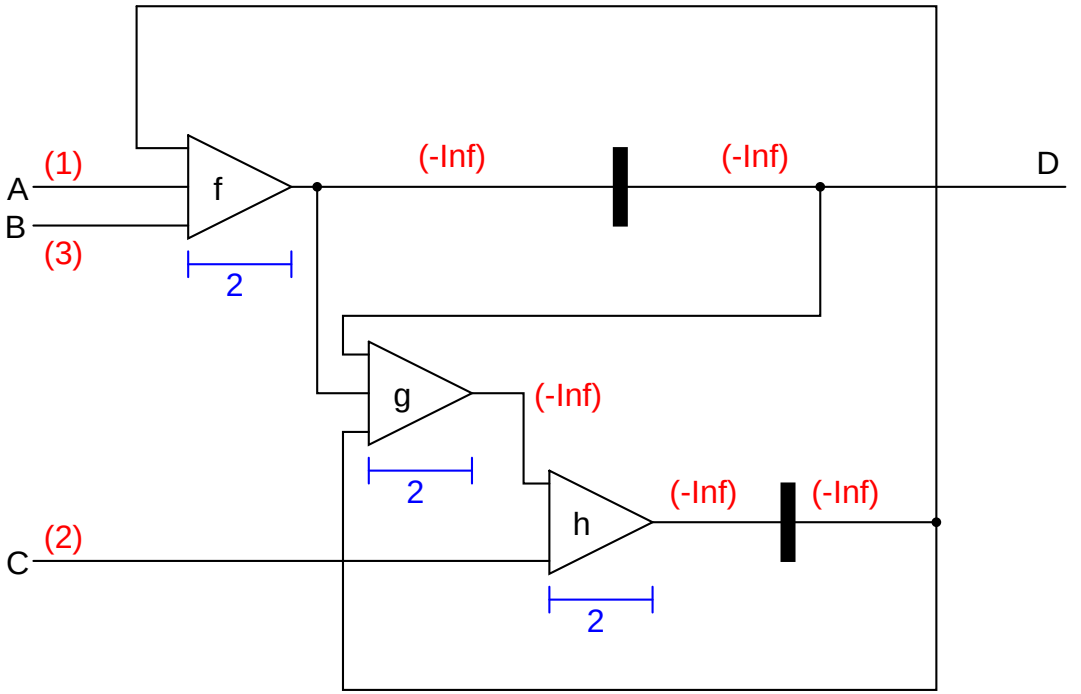
desired period : 3

multiplexer delay : 1

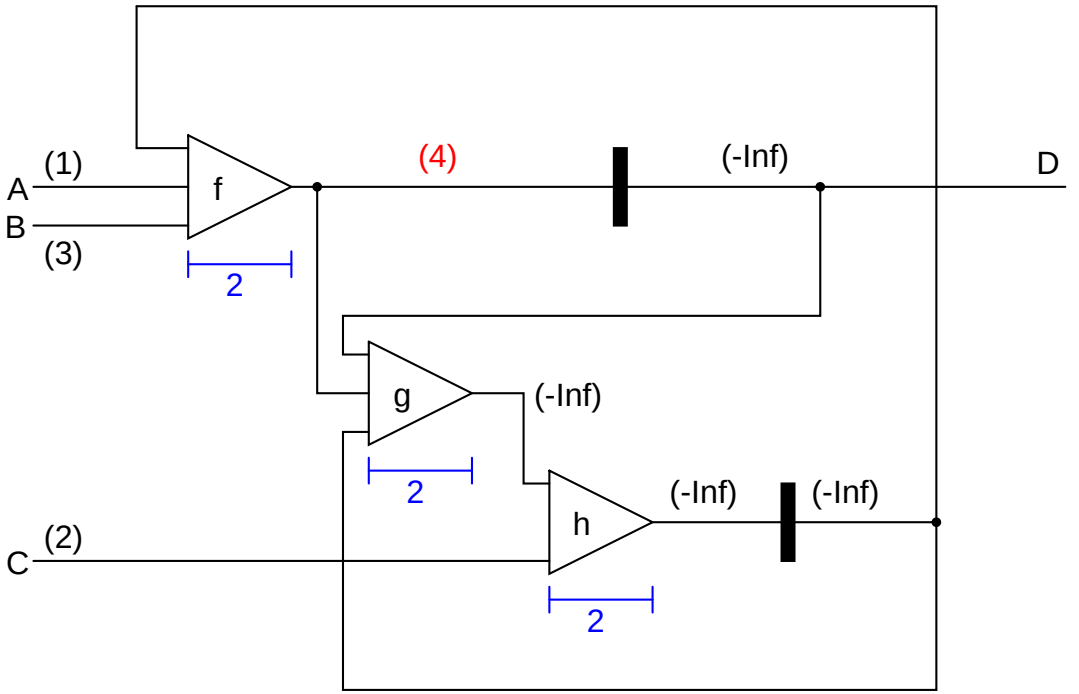
input arrival times: A=1 B=3 C=2

output required time(s): D=3

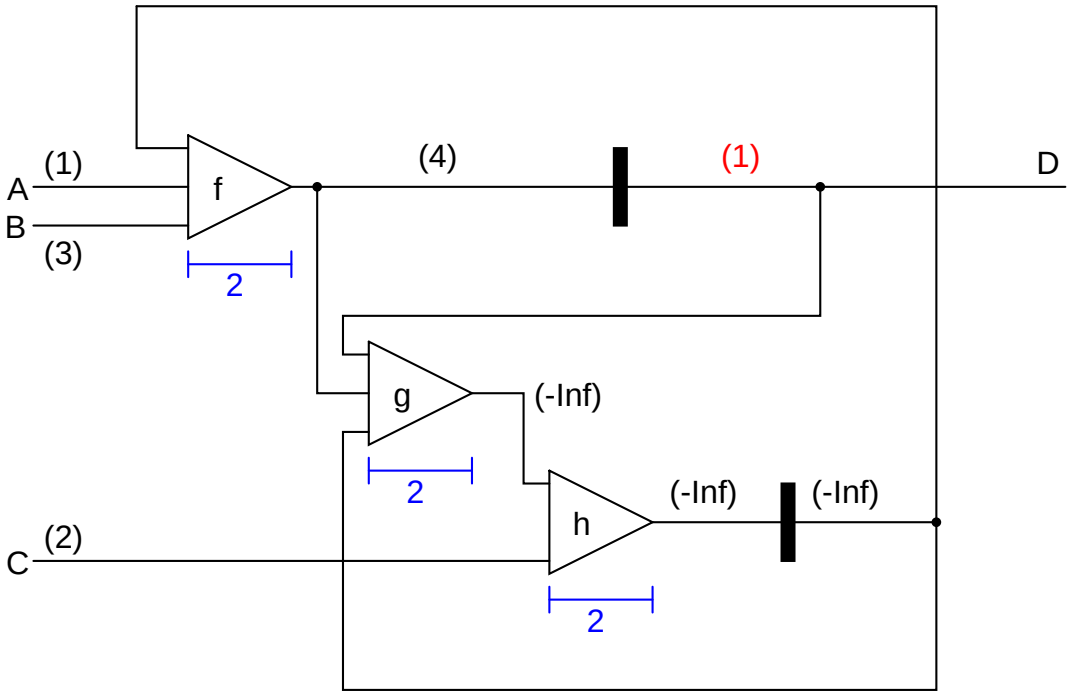
Bellman-Ford : initialization



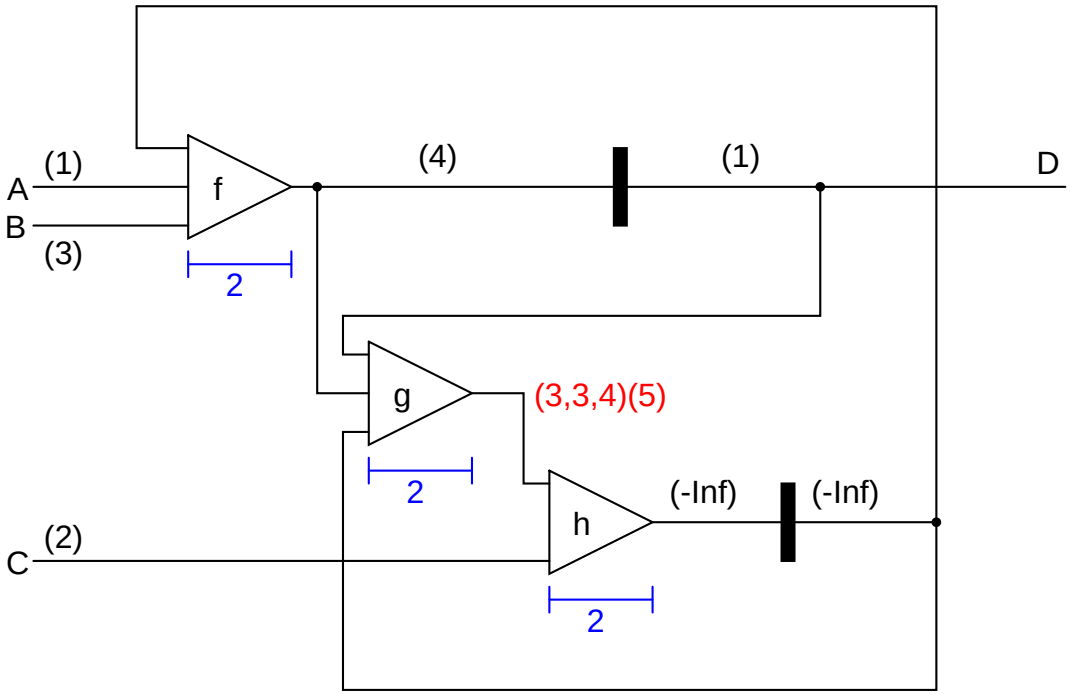
Bellman-Ford : starting relaxation



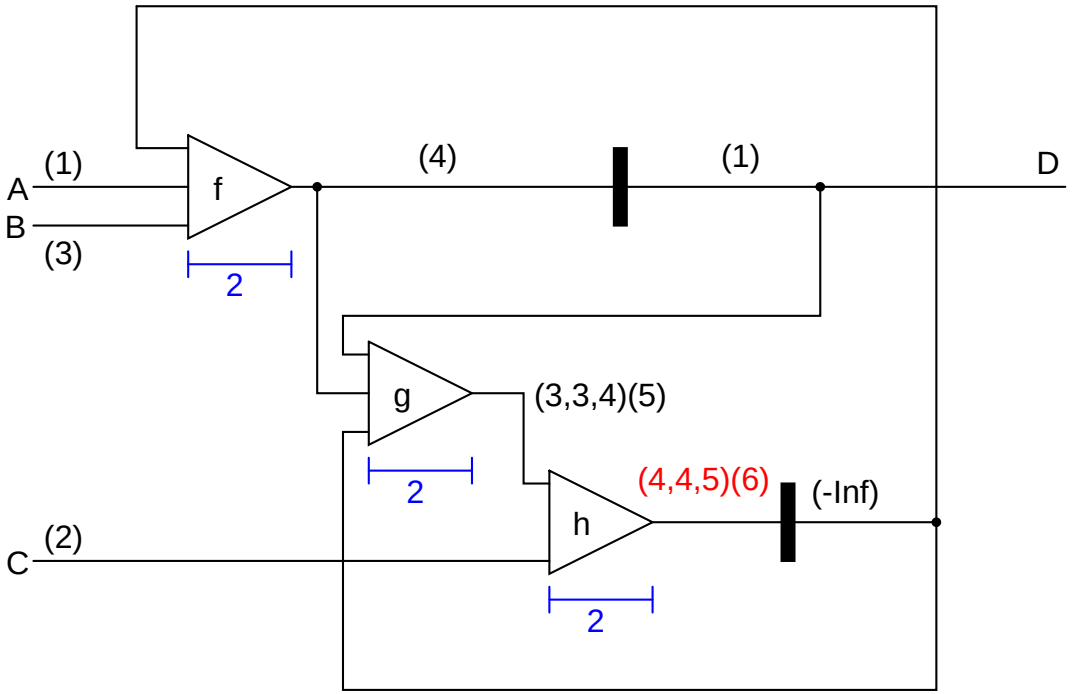
Bellman-Ford : relaxing ...



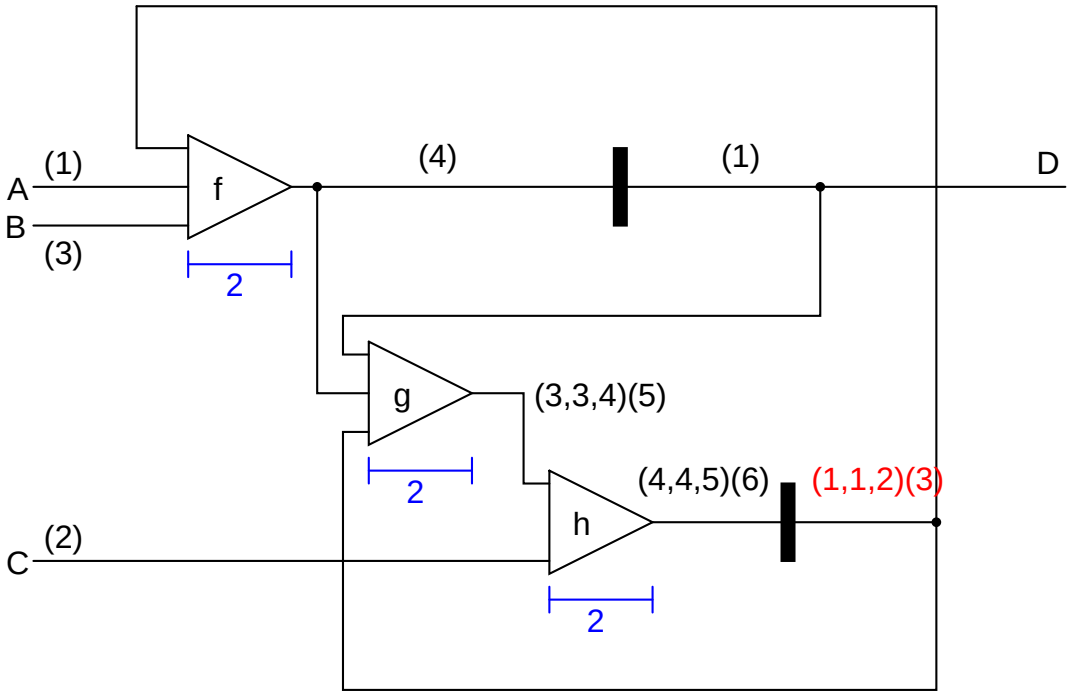
Bellman-Ford : relaxing ...



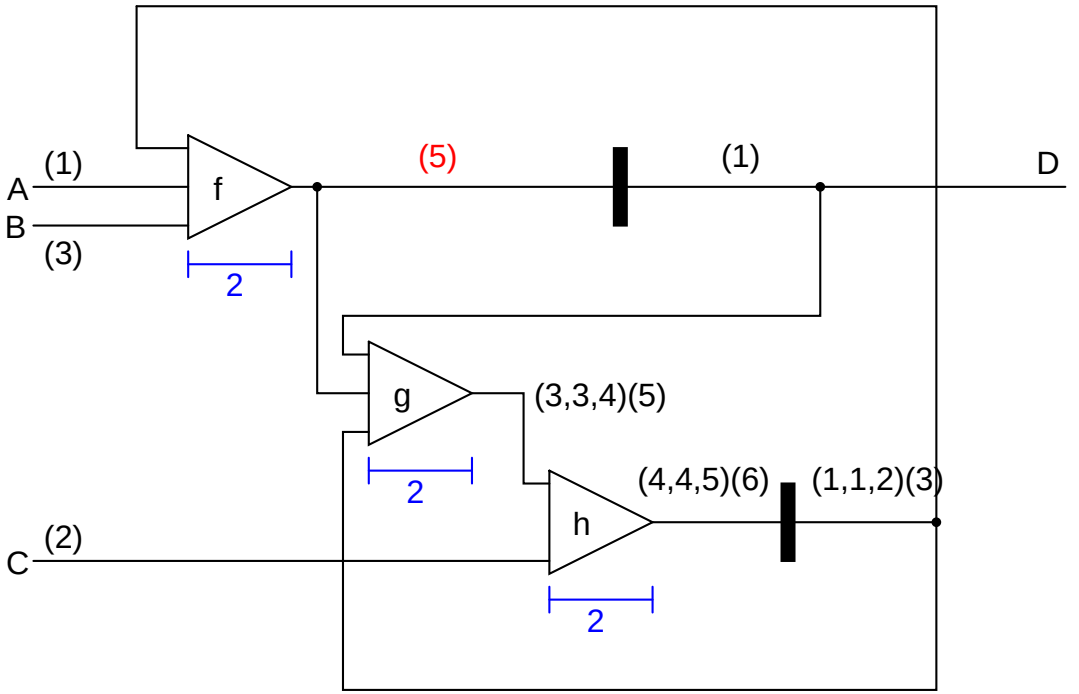
Bellman-Ford : relaxing ...



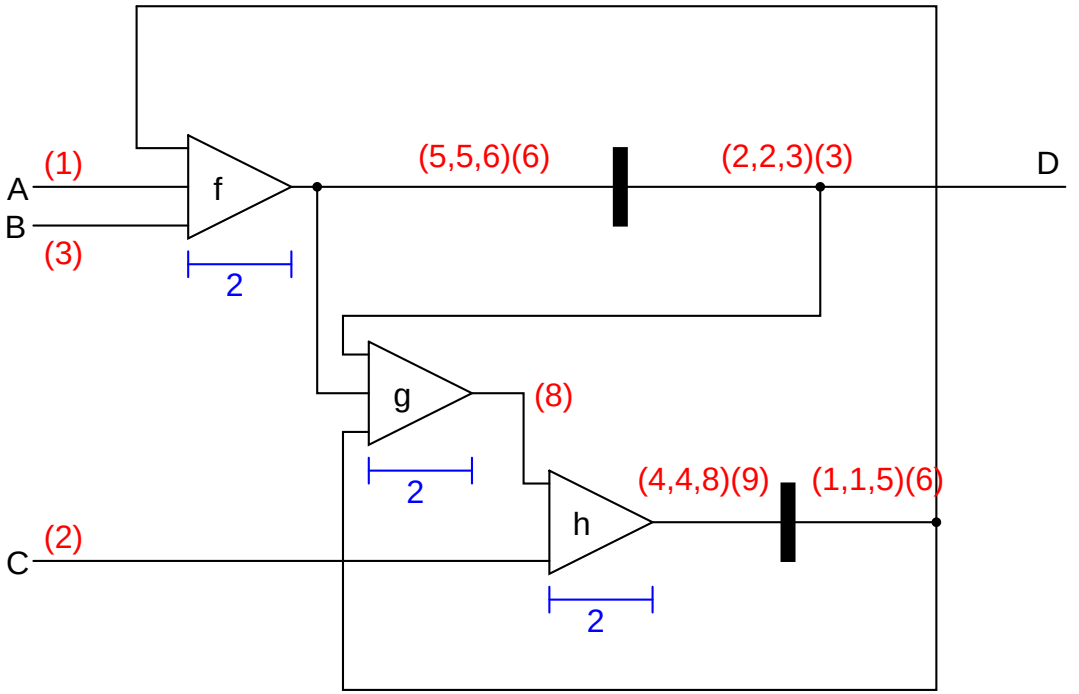
Bellman-Ford : relaxing ...



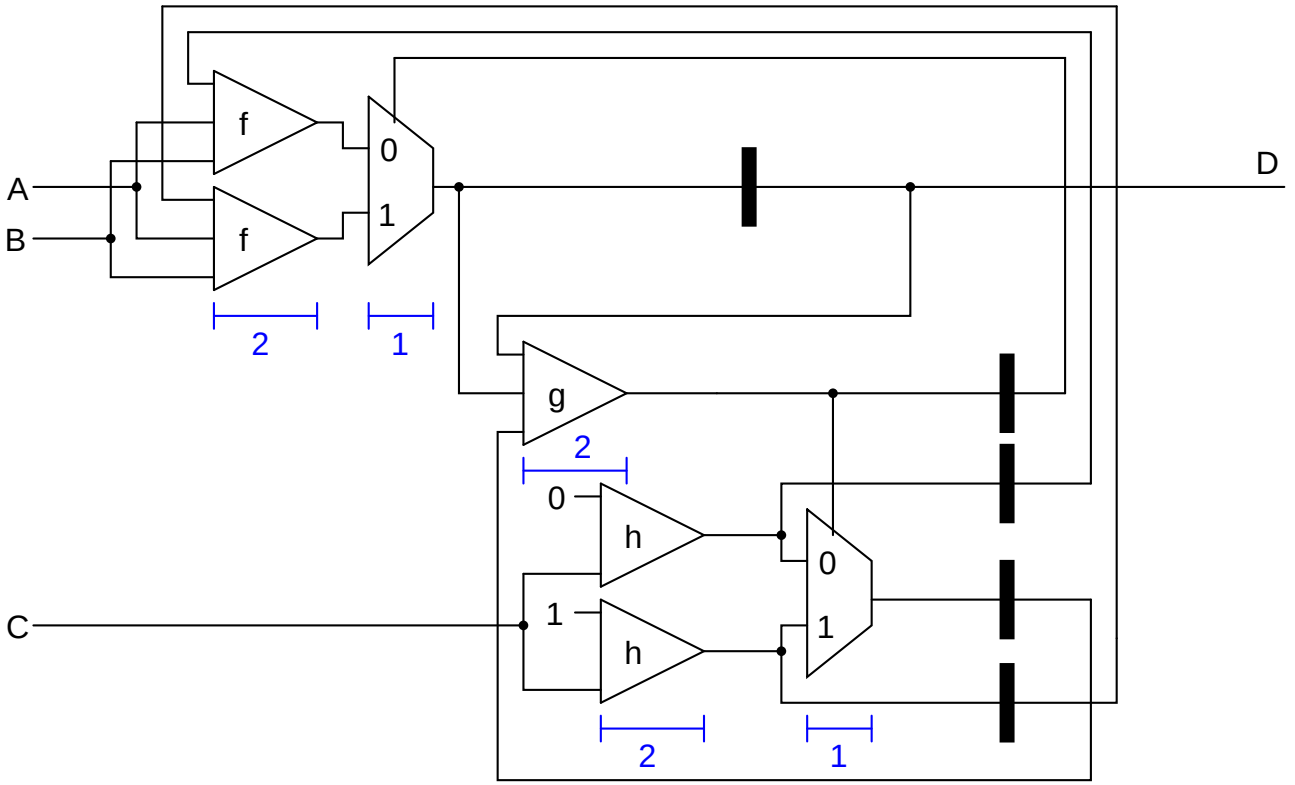
Bellman-Ford : relaxing ...



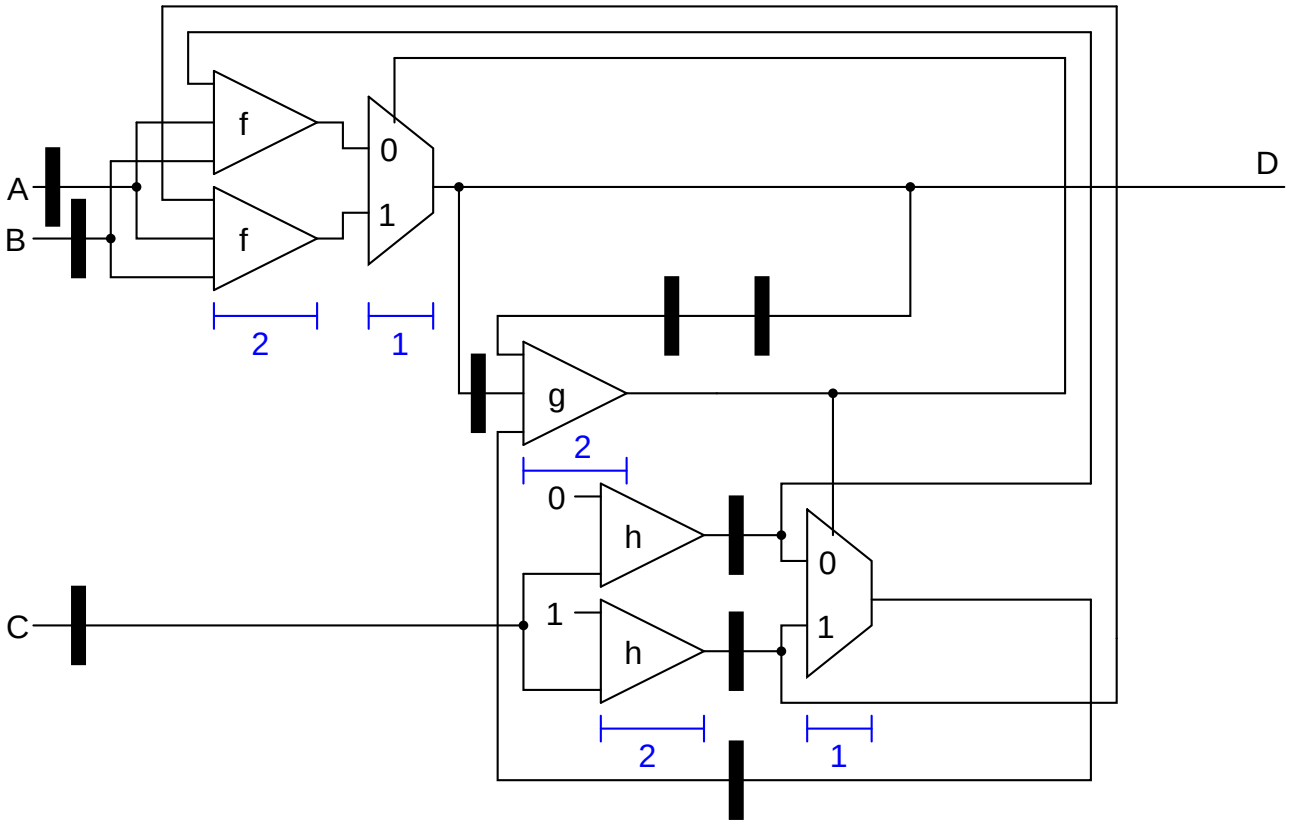
Bellman-Ford : fi x point found



ShannonTransform

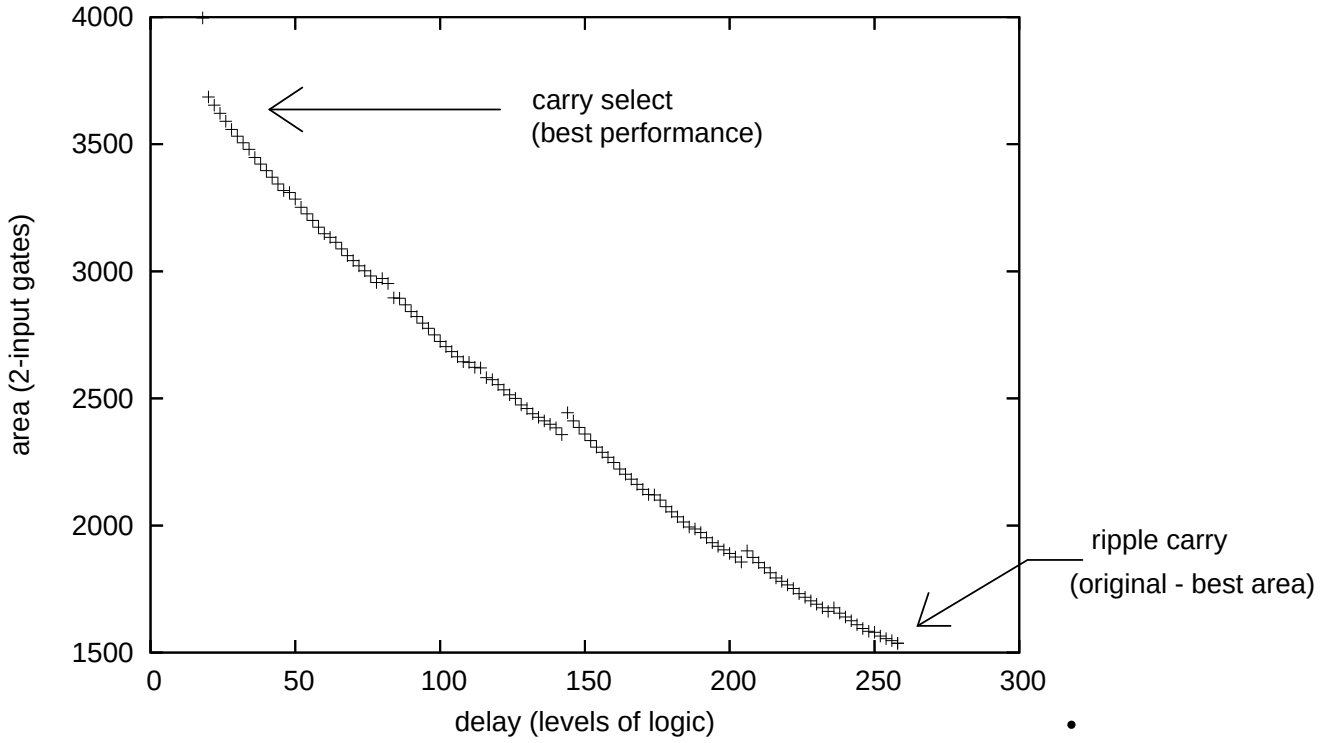


Retiming



Performance / area tradeoff —sanity check

Ripple carry adders $\rightarrow log(n)$ delay carry select adders



add128. all above 120 points computed in 22s.

ISCAS89 sequential benchmarks

	reference		retimed		Sh. + ret.		time	speed	area
	period	area	period	area	period	area	(s)	up	penalty
s510	8	184	8	184	8	184	0.5		
s641	11	115	11	115	9	122	1.1	22%	6%
s713	11	118	11	118	10	121	0.9	10%	3%
s820	7	206	7	206	7	206	0.5		
s832	7	217	7	217	7	217	0.4		
s838	10	154	10	154	8	162	2.6	25%	5%
s1196	9	365	9	365	9	365	0.6		
s1423	24	408	21	408	13	460	3.8	61%	12%
s1488	6	453	6	453	6	453	0.7		
s1494	6	456	6	456	6	456	0.8		
s9234	11	662	8	656	8	684	6.7		
s13207	14	1382	11	1356	9	1416	18.0	22%	4%
s38417	14	7706	14	7652	13	7871	113	7%	3%