# Optimal resource assignment in workflows for maximizing cooperation

Akhil Kumar[1], R.M. Dijkman[2], Minseok Song[3]

[1]Smeal College of Business, Penn State University, USA
AkhilKumar@psu.edu
[2]Eindhoven University of Technology, The Netherlands
r.m.dijkman@tue.nl
[3]Ulsan National Institue of Science and Technology, South Korea
msong@unist.ac.kr

PENNSTATE
1855

# Outline

- Motivation
- Basic model
- Optimal vs. Greedy algorithm
- Computing compatibility
- Discussion and extensions
- Conclusions

# Need for cooperation

"We found that patients whose surgical teams exhibited less teamwork behaviors were at a higher risk for death or complications." [ref. 10]

A workflow is also a team process!

# Handoffs


Bedside Nurse to Nurse handoff

Physician to physician handoff

# Soft vs. hard handoff

H*ard handoff* : no further interaction between the two actors of different tasks is required.

*Soft handoff*: the two actors of different tasks may still need to interact later for queries and clarifications (handoff ≠ hands off!)

Thus, in practice "there is a series of *overlapping* and *nested* roles and responsibilities."[Leach 2009, Performance of Surgical teams]

*Compatibility* is a measure of the degree to which actors cooperate with one another in a workflow.

*Degree of cooperation* between a pair of tasks (say, on a 0-1 scale) reflects the amount of cooperation needed between the tasks

Examples of soft handoffs: Medical treatment, software development

PENNSTATE

# A claims process

Receive claim

(H1)

Customer representative

Review claim

(H2)

Reviewer

Evaluate claim

(H3)

Claim Evaluator

Approve claim

(H4)

Manager

Pay claim

Accounts officer

H1, H2, H3 are soft handoffs, e.g.

- H1: A reviewer may refer back to the customer representative for missing or unrecorded information
- H2: An evaluator may contact the reviewer for some policy details
- H3: A manager may contact the evaluator and the reviewer for additional information before approving the payment

# Compatibility metric

Our metric for compatibility within a workflow is:

$$Total\ Compatibility = \sum_{\forall(u1,u2,t1,t2)} fit_{u1,u2,t1,t2} * coop_{t1,t2} * cweight_{u1,u2}$$

$$Average\ Compatibility = \frac{Total\ Compatibility}{\sum_{t1,t2} coop_{t1,t2}}$$

Where

$$fit_{u1,u2,t1,t2}: \begin{cases} 1\ if\ actor\ u1, u2\ perform\ tasks\ t1, t2\ respectively \\ 0, otherwise \end{cases}$$

$$coop_{t1,t2}: \begin{cases} 1, if\ cooperation\ required\ between\ tasks\ t1\ and\ t2 \\ 0, otherwise \end{cases}$$

$cweight_{u1,u2}$: $compatibilty\ of\ actors\ u1, u2$ on a continuous scale of 0-1

# Example of compatibility matrix

## Actor-actor compatibility

(degree of compatibility between a pair of actors (0-1 scale)

| Role > (Task) > | Cust. Rep (receive) | | Reviewer (validate) | | Evaluator (settle) | | Manager (approve) | | Accounts (pay) | |
|---|---|---|---|---|---|---|---|---|---|---|
| | John | Mary | Beth | Sue | Mike | Jim | Jen | Pat | Mark | Lin |
| **John** | 1.0 | 0.9 | 0.8 | 0.1 | 0.8 | 0.3 | 0.9 | 0.3 | 0.4 | 0.2 |
| **Mary** | 0.9 | 1.0 | 0.3 | 0.7 | 0.2 | 0.8 | 0.9 | 0.2 | 0.1 | 0.8 |
| **Beth** | 0.8 | 0.3 | 1.0 | 0.8 | 0.3 | 0.8 | 0.7 | 0.3 | 0.2 | 0.9 |
| **Sue** | 0.1 | 0.7 | 0.8 | 1.0 | 0.9 | 0.6 | 0.4 | 0.6 | 0.8 | 0.4 |
| **Mike** | 0.8 | 0.2 | 0.3 | 0.9 | 1.0 | 0.9 | 0.3 | 0.9 | 0.8 | 0.1 |
| **Jim** | 0.3 | 0.8 | 0.8 | 0.6 | 0.9 | 1.0 | 0.8 | 0.1 | 0.3 | 0.9 |
| **Jen** | 0.9 | 0.9 | 0.7 | 0.4 | 0.3 | 0.8 | 1.0 | 0.8 | 0.7 | 0.3 |
| **Pat** | 0.3 | 0.2 | 0.3 | 0.6 | 0.6 | 0.9 | 0.8 | 1.0 | 0.1 | 0.8 |
| **Mark** | 0.4 | 0.1 | 0.2 | 0.8 | 0.8 | 0.3 | 0.7 | 0.1 | 1.0 | 0.9 |
| **Lin** | 0.2 | 0.8 | 0.9 | 0.4 | 0.1 | 0.9 | 0.3 | 0.8 | 0.9 | 1.0 |

# Example of cooperation matrix

PARTIAL cooperation – SOME task pairs require cooperation

| | Receive | Validate | Settle | Approve | Pay |
|---|---|---|---|---|---|
| **Receive** | – | 1 | 1 | 0 | 0 |
| **Validate** | 1 | – | 1 | 1 | 0 |
| **Settle** | 1 | 1 | – | 1 | 0 |
| **Approve** | 0 | 1 | 1 | – | 1 |
| **Pay** | 0 | 0 | 0 | 1 | – |

FULL cooperation – ALL task pairs require cooperation

| | Receive | Validate | Settle | Approve | Pay |
|---|---|---|---|---|---|
| **Receive** | – | 1 | 1 | 1 | 1 |
| **Validate** | 1 | – | 1 | 1 | 1 |
| **Settle** | 1 | 1 | – | 1 | 1 |
| **Approve** | 1 | 1 | 1 | – | 1 |
| **Pay** | 1 | 1 | 1 | 1 | – |

# Random vs. optimal cooperation

Partial cooperation

Assignment 1 (random):

cust. rep: John ; Reviewer: Sue; Evaluator: Jim; Manager: Pat; accounts officer: Mark

Average compatibility = (0.1 + 0.3 + 0.6 + 0.6 + 0.1 + 0.1)/6 = **0.3**

Assignment 2 (optimal):

cust rep: Mary; Reviewer: Beth; Evaluator: Jim; Manager: Jen; accounts officer: Mark

Average compatibility = (0.3 + 0.8 + 0.8 + 0.7 + 0.8 + 0.7)/6 = **0.68**

Full cooperation

Assignment 3 (random):

cust rep: Mary ; Reviewer: Sue; Evaluator: Jim; Manager: Jen; accounts officer: Lin

Average compatibility =

(0.7 + 0.8 + 0.9 + 0.8 + 0.6 + 0.4 + 0.4 + 0.8 + 0.9 + 0.3)/10= **0.66**

Assignment 4 (optimal):

cust rep: Mary; Reviewer: Beth; Evaluator: Jim; Manager: Jen; accounts officer: Lin

Average compatibility =

(0.3 + 0.8 + 0.9 + 0.8 + 0.8 + 0.7 + 0.9 + 0.8 + 0.9 + 0.3)/10= **0.72**

# Optimal assignment Model - OWA

Model OWA

$$Minimize \sum_{u1,u2,t1,t2} fit_{u1,u2,t1,t2} * (1 - cweight_{u1,u2})$$

(Minimize *non*-compatibility)

Subject to:

$\sum_u does_{u,t} = 1, \ \forall t$ *At least one actor must do each task*

$does_{u,t} \le cando_{u,t}$ *Only a qualified actor can do a task*

$does_{u1,t1} + does_{u2,t2} - fit_{u1,u2,t1,t2} \le 1, \ \forall\, t1, t2 \text{ where } coop(t1,t2) = 1$

***Forces fit variable to 1 when cooperation required between t1, t2***

PENNSTATE 1855

# Greedy Heuristic

```
Algorithm Greedy_Coop
        Input: coop[][],cweight[][], cando[][]
        Output: assign[]
      for each (task t1 = 1,…, num_tasks)  For each task t1
        for each (u1  cando[t1])           For each user
         for each (task t2 = t1+1,…, num_tasks) For task t2
      if (coop(t1,t2)&& not(assign[t1])&& not(assign[t2]):
         score(u1)=score(u1) + max(cweight(u1,u2),
                                          ∀ u2 ϵ cando[t2])
          end for
        end for
        assign[t1] = u*,    Make (u1,t1) greedy assignment
            s.t. score[u*],u*= max(score[u], u ϵ cando[t1])
        for each (task t2 = t1+1,…, num_tasks)
       if (coop(t1,t2)&& not(assign[t2]):
         assign[t2] = u2*, s.t. (u2,t2) greedy assignment
        cweight[u*,u2*]=max(cweight[u*,u2], u2 ϵ cando[t2])
      end for
       return(assign[])
```

# Optimal vs. Greedy Results

## 10 tasks, 20 actors

| | Avg. Compat. | | % gap |
|---|---|---|---|
| Case | Greedy | Opt. | |
| 1. | 0.656 | 0.700 | 6.29 |
| 2 | 0.650 | 0.759 | 14.36 |
| 3. | 0.669 | 0.760 | 11.97 |
| 4. | 0.653 | 0.785 | 16.82 |
| 5. | 0.591 | 0.740 | 20.14 |
| 6. | 0.615 | 0.737 | 16.55 |
| 7. | 0.461 | 0.597 | 22.78 |
| 8. | 0.550 | 0.761 | 27.73 |
| 9. | 0.466 | 0.780 | 40.26 |
| 10. | 0.615 | 0.730 | 15.75 |
| Avg. | 0.593 | 0.735 | 19.32 |

## 20 tasks, 40 actors

| | Avg. Compat. | | % gap |
|---|---|---|---|
| Case | Greedy | Opt. | |
| 1. | 0.615 | 0.684 | 10.09 |
| 2 | 0.568 | 0.717 | 20.78 |
| 3. | 0.557 | 0.607 | 8.24 |
| 4. | 0.605 | 0.759 | 20.29 |
| 5. | 0.608 | 0.712 | 14.61 |
| 6. | 0.596 | 0.771 | 22.70 |
| 7. | 0.567 | 0.734 | 22.75 |
| 8. | 0.556 | 0.718 | 22.56 |
| 9. | 0.570 | 0.691 | 17.51 |
| 10. | 0.619 | 0.720 | 14.03 |
| Avg. | 0.586 | 0.711 | 17.36 |

In both cases, there is an almost 20% gap between Optimal and Greedy

PENNSTATE
1855

# Computing the compatibility matrix

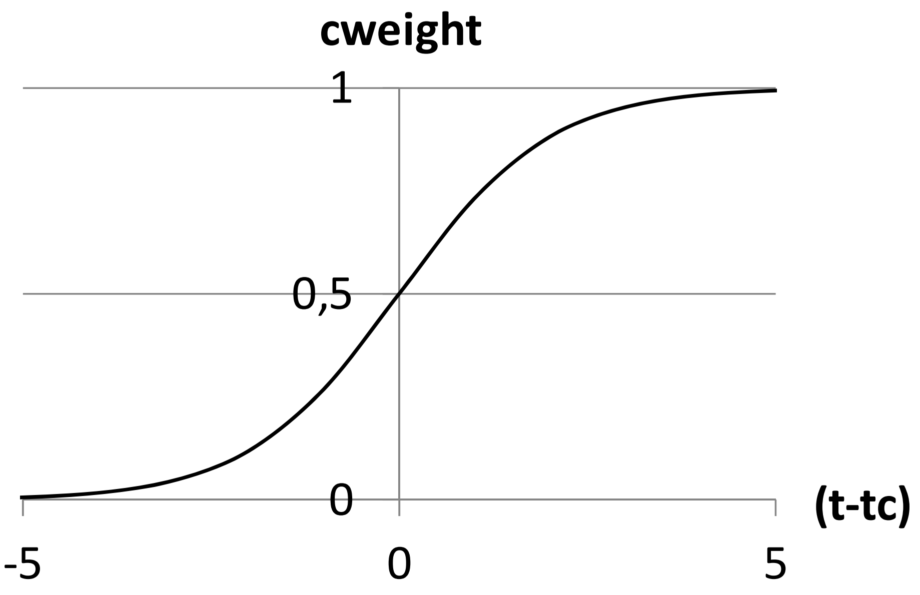
<u>Sigmoid function</u>

<u>Given</u>: two actors $\boldsymbol{u_1}$ and $\boldsymbol{u_2}$; the <u>average</u> throughput time $\boldsymbol{t}$ of the process; and the average throughput time $\boldsymbol{tc}$ of the process for execution traces in which $u_1$ and $u_2$ collaborated:

$$cweight_{u1,u2} = \frac{1}{1 + e^{-k(t-tc)}}$$

**K** is a parameter

Relation between $(t - tc)$ and *cweight* in a collaboration



# Example to show compatibility matrix computation

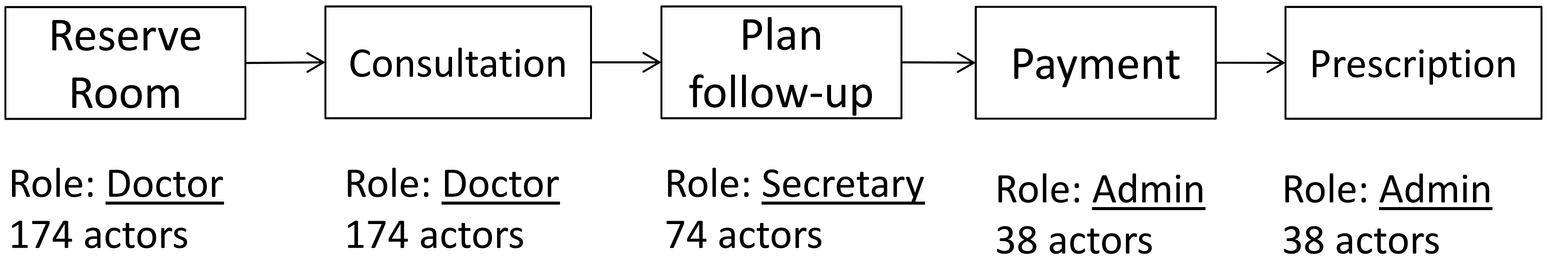
| (a) Execution traces | | | | |
|---|---|---|---|---|
| Trace | Receive | Validate | Settlement | t |
| 1 | John | Mary | Mike | 8 |
| 2 | John | Beth | Mike | 10 |
| 3 | John | Mary | Mike | 9 |

| (b) Average throughput times | | | | |
|---|---|---|---|---|
| | John | Mary | Beth | Mike |
| John | 9 | 8.5 | 10 | 9 |
| Mary | 8.5 | 9 | – | 8.5 |
| Beth | 10 | – | 9 | 10 |
| Mike | 9 | 8.5 | 10 | 9 |

| (c) Compatibility matrix | | | | |
|---|---|---|---|---|
| | John | Mary | Beth | Mike |
| John | 0.5 | 0.6 | 0.3 | 0.5 |
| Mary | 0.6 | 0.5 | – | 0.6 |
| Beth | 0.3 | – | 0.5 | 0.3 |
| Mike | 0.5 | 0.6 | 0.3 | 0.5 |

# Empirical evaluation

Site: Seoul National University Bundang Hospital, South Korea
Process: Doctor consultation
Traces: 4,446 execution traces




Analysis approach

- We focused on collaborations that occurred more than 20 times
- Focused on collaborations in the third, fourth and fifth steps of the process. (These administrative tasks are less case specific)
- 1,717 execution traces, 35 pairs of collaborations
- Shapiro-Wilk test showed that the data was *not* normally distributed.
- Kruskal-Wallis test found that collaborations differed significantly, at 5% level

# Computing compatibility matrix

- Used the Sigmoid function to determine *cweight*
- Actor names are coded such as EIC, CDCJJ, etc. for anonymity
- The full compatibility matrix has 286 x 286 cells
- Average throughput time of optimal assignment is 6 minutes, which is a strong improvement over the overall average throughput time of 42.9 minutes (from c-matrix)

| | EIC | CDCJJ | CHBAB | CEFGG | … |
|---|---|---|---|---|---|
| EIC | 0.50 | 0.99 | – | – | … |
| CDCJJ | 0.99 | 0.50 | 0.25 | 0.75 | … |
| CHBAB | – | 0.25 | 0.50 | – | … |
| CEFGG | – | 0.75 | – | 0.50 | … |
| … | … | … | … | … | … |

PENNSTATE

# Results

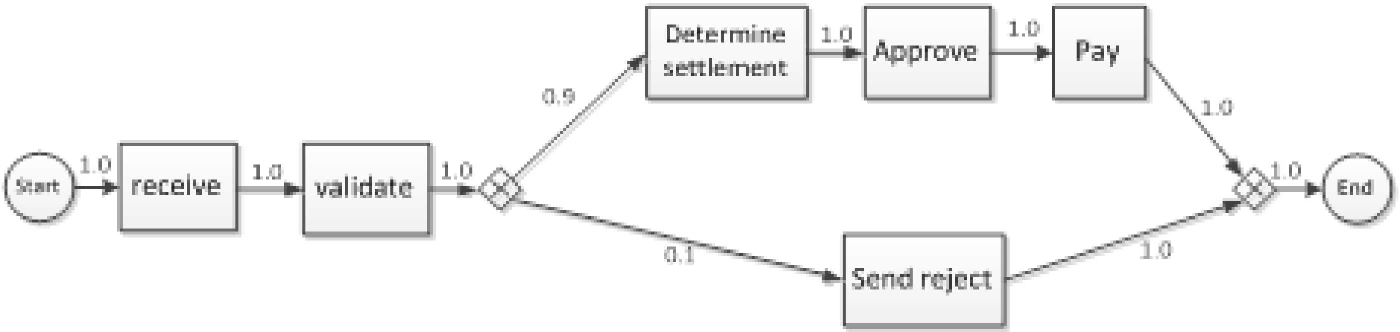
| Scenario | Avg. Throughput Times |
|---|---|
| Overall avg. throughput time (c-matrix) | 42.9 minutes |
| Optimal avg. throughput time (c-matrix) | 6 minutes |
| "Best assignment" from 10 or more execution traces | 23.7 minutes |
| Another assignment from 10 or more execution traces | 19.7 minutes |
| Best assignment from 20 or more execution traces | 26.9 minutes |

# Extensions

- Varying degrees of required cooperation

$$Minimize \sum_{\forall u1,u2,t1,t2} coop_{t1,t2} fit_{u1,u2,t1,t2} * (1 - cweight_{u1,u2})$$

- Multiple paths with path transition probabilities




- Optimization of cost, time, resource, other constraints

# Related work

- (Seidmann and Sundararajan, 1997) If knowledge intensity and customization level are high, smooth handoff is critical
- (Tjosvold, D., et al. 1999): cooperative behavior affects service quality and performance
- (Wolf, 2009): modeling and scheduling of clinical pathways
- (Song and Aalst, 2008): organizational mining to discover organizational models (another way to determine compatibility)
- (Mansar and Reijers, 2005): cooperation among actors has implications for best practices in business process redesign
- (Reijers, et al., 2007): assigning work in emergency situations using threshold models
- (Reijers, et al., 2008): geographical dispersion of workers hurts performance
- (Berkenstadt, et al., 2008): handoff protocol in critical care

# Discussion

- Workflow as a team process!
- All handoffs between tasks are not *hard*
- Cooperative behavior affects service quality and performance
- Need for smooth handoff related to knowledge intensity and level of customization
- Average time for task t by actor u =

  $f(\text{capability}(u,t), \text{compatibility}(u, u_i), \text{trust}(u,u_i))$
- Implications for best practices in process design in terms of task boundaries
- Handoff protocols in shift handoffs

PENNSTATE 1855

# Conclusions and Future work

- When handoffs are soft, cooperative behavior can affect performance
- We developed an optimization model for work assignment in such a context and first solved it for simulated data
- Then, we tested model on real data from healthcare
- Showed that throughput time varies considerably as a function of assignments
- Plan to run tests with more datasets
- Better ways to model soft handoffs
- Find ways to describe cooperative behavior in workflow modeling languages
- Study how to model handoffs between teams with constraints on team composition