CrowdManager

Combinatorial Allocation and Pricing of Crowdsourcing Tasks with Time Constraints

Patrick Minder, Sven Seuken, Abraham Bernstein, Mengia Zollinger
“Guess” the price per task

Post the task on Mechanical Turk

Wait
Crowdsourcing under Time-Constraints

Applications

- **Text Shortening** (e.g., Bernstein et al. 2010)
- **Real-Time On-line Services** (e.g., Bigham et al. 2010)
- **Text Translation** (e.g., Minder & Bernstein 2012)
- **Fraud-Detection**
Crowdsourcing under Time-Constraints

Challenges

Crowd Latency vs. Time-Constraints

Quality Management

Dynamic Pricing
Related Work

• **Retainer Model [Bernstein et al. 2011]**
  - Pre-recruit workers
  - Price per task is fixed ex-ante

• **On-line Pricing Mechanism [Singer 2011]**
  - Maximize number of solved tasks under budget
  - Mechanism-design inspired approach
  - No other constraints considered
Contributions

• CrowdManager’s framework architecture
• A mechanism for the combinatorial allocation and pricing of crowdsourcing tasks under budget, completion time, and quality constraints
• Initial evaluation incorporating a simulation
Single requestor with work package $W$ containing a set of $m$ similar tasks

Requestor has a budget ($B$), completion time ($T$), and quality ($Q$) constraints

Requestor has a quasi-linear utility $U = B - C$ if all $m$ task get solved within $T$ and under $Q$, and $U = -C$ otherwise
Formal Model - Worker

- Each worker has private costs $c > 0$ for solving a task
- A worker wants to solve at most $j > 0$ tasks
- We rate a worker’s qualification level $q \in [0, 1]
- A estimate a workers completion time $t > 0$ for solving a task
Limitations

• Assumptions:
  - Workers can’t fake completion time and quality
  - No worker leaves the Retainer

• Beyond the scope of this paper
  - Recruitment process
  - Moral hazard
  - Evaluation of quality
Outline

1 Formal Model
2 Platform
3 Mechanism
4 Evaluation
5 Discussion
Time-Constrained Text Translation

- Translate 10 pages from German to English
- Within the next 20 minutes
- In good quality
- As cheap as possible, but for at most 10$
Time-Constrained Text-Translation

Walkthrough

Requestor

Notify \(\rightarrow\) Translate \(\rightarrow\) Wait \(\rightarrow\) Received

CrowdManager

ClickWorker \(\rightarrow\) CrowdFlower \(\rightarrow\) MTurk \(\rightarrow\) Elance
Time-Constrained Text-Translation

Walkthrough

Requestor:
- Notify
- Translate
- Wait
- Received

Rewrite 20 sentences max. 2$ within 4 minutes in "good quality"

CrowdManager:
- CrowdManager Application Interface
- Kernel
- Recruitment Interface
- Requestor Application
- ClickWorker
- CrowdFlower
- MTurk
- Elance

Crowd:
Time-Constrained Text-Translation

Walkthrough

- Notify
- Translate
- Wait
- Received

CrowdManager

- ClickWorker
- CrowdFlower
- MTurk
- Elance
Time-Constrained Text-Translation

Walkthrough

Requestor

Notify ➔ Translate ➔ Wait ➔ Received

CrowdManager

Job Proposal

Crowd

ClickWorker ➔ CrowdFlower ➔ MTurk ➔ Elance
Time-Constrained Text-Translation

Walkthrough

Requestor

Notify → Translate → Wait → Received

CrowdManager

ClickWorker  CrowdFlower  MTurk  Elance
Time-Constrained Text-Translation

Walkthrough

Requestor

Notify → Translate → Wait → Received

CrowdManager

ClickWorker, CrowdFlower, MTurk, Elance

Crowd
Time-Constrained Text-Translation

Walkthrough

Requestor

Notify → Translate → Wait → Received

CrowdManager

Tasks

Crowd

ClickWorker, CrowdFlower, MTurk, Elance
Time-Constrained Text-Translation

Walkthrough

Requestor

- Notify
- Translate
- Wait
- Received

CrowdManager

ClickWorker  CrowdFlower  MTurk  Elance

Crowd
CrowdManager's Bidding Interface

We have 25 translation tasks to complete. Please solve the following sample task. We will use your answer to determine whether you can participate in this translation exercise. Afterwards, please specify how many of such translation tasks you would like to solve, and which minimum wage per task you would accept.

Example Task
Please improve the following sentence by correcting grammatical errors and making the sentence more comprehensible.

"Wikipedia is a Founded in January 2001 free online encyclopedia in many languages."

Answer:

Wikipedia is a free online encyclopedia that is available in many languages and was founded in January 2001.

How many of those tasks do you want to solve? 4

What is the minimum wage per task you would accept? 1$
Time-Constrained Text-Translation

Walkthrough

Requestor

Notify → Translate → Wait → Received

CrowdManager

Crowd

ClickWorker, CrowdFlower, MTurk, Elance
Time-Constrained Text-Translation

Walkthrough

Requestor
- Notify
- Translate
- Wait
- Received

CrowdManager

Crowd
- ClickWorker
- CrowdFlower
- MTurk
- Elance
Time-Constrained Text-Translation

Walkthrough

Requestor

Notify → Translate → Wait → Received

CrowdManager

Crowd

ClickWorker  CrowdFlower  MTurk  Elance

Retainer

Tasks  Results
Time-Constrained Text-Translation

Walkthrough

Requestor

Notify → Translate → Wait → Received

CrowdManager

ClickWorker  CrowdFlower  MTurk  Elance

Crowd
Time-Constrained Text-Translation

Walkthrough

- Requestor
  - Notify
  - Translate
  - Wait
  - Received

- CrowdManager
  - Crowd
    - ClickWorker
    - CrowdFlower
    - MTurk
    - Elance
Time-Constrained Text-Translation

Walkthrough

Requestor

Notify → Translate → Wait → Received

CrowdManager

Result

Crowd

ClickWorker • CrowdFlower • MTurk • Elance
Time-Constrained Text-Translation

Walkthrough

Requestor

Notify -> Translate -> Wait -> Received

CrowdManager

Tasks

Results

Crowd

ClickWorker  CrowdFlower  MTurk  Elance
1 Formal Model
2 Platform
3 Mechanism
4 Evaluation
5 Discussion
Allocation and Pricing Mechanism

Algorithm

Require: \( I, W, B, T, Q \)

\( \hat{\theta} = \text{runProcurementAuction}(I, \text{QualificationTest}, m) \)

\( x = \text{allocationMechanism}(\hat{\theta}, W, T, Q) \)

if \( x \) is feasible then

\( p = \text{paymentMechanism}(\hat{\theta}, x) \)

else

return no completion time feasible allocation found

end if

\( \text{costs} = \sum_{i \in I} p_i \cdot x_i \)

if \( \text{costs} \leq B \) then

return \( x, p \)

else

return no budget-feasible solution found

end if
Allocation and Pricing Mechanism

Algorithm

Require: $I, W, B, T, Q$

\[ \hat{\theta} = \text{runProcurementAuction}(I, \text{QualificationTest}, m) \]

\[ x = \text{allocationMechanism}(\hat{\theta}, W, T, Q) \]

if $x$ is feasible then

\[ p = \text{paymentMechanism}(\hat{\theta}, x) \]

else

return no completion time feasible allocation found

end if

\[ \text{costs} = \sum_{i \in I} p_i \cdot x_i \]

if $\text{costs} \leq B$ then

return $x, p$

else

return no budget-feasible solution found

end if
Algorithm

**Require:** $I, W, B, T, Q$

$$\hat{\theta} = \text{runProcurementAuction}(I, \text{QualificationTest}, m)$$

$$x = \text{allocationMechanism}(\theta, W, T, Q)$$

**if** $x$ is feasible **then**

$$p = \text{paymentMechanism}(\hat{\theta}, x)$$

**else**

**return** no completion time feasible allocation found

**end if**

$$\text{costs} = \sum_{i \in I} p_i \cdot x_i$$

**if** costs $\leq B$ **then**

**return** $x, p$

**else**

**return** no budget-feasible solution found

**end if**
maximize social welfare

\[
\min_{x_1, \ldots, x_n} \sum_{i=1}^{n} \hat{C}_i x_i
\]
Allocation Mechanism

Combinatorial Optimization Problem

maximize social welfare

All Tasks Assigned

\[ \text{min} \sum_{i=1}^{n} \hat{C}_i x_i \]

\[ \sum_{i=1}^{n} x_i = m \]
Allocation Mechanism

Combinatorial Optimization Problem

maximize
social welfare

All Tasks Assigned

Time Constraint

\[
\begin{align*}
\text{min} & \quad \sum_{i=1}^{n} \hat{C}_i x_i \\
\text{s.t.} & \quad \sum_{i=1}^{n} x_i = m \\
& \quad x_i \cdot \hat{t}_i \leq T, \quad \forall i \in I
\end{align*}
\]
**Allocation Mechanism**

**Combinatorial Optimization Problem**

Maximize social welfare

\[
\min_{x_1, \ldots, x_n} \sum_{i=1}^{n} \hat{C}_i x_i
\]

Subject to

- All Tasks Assigned:
  \[
  \sum_{i=1}^{n} x_i = m
  \]

- Time Constraint:
  \[
  x_i \cdot \hat{t}_i \leq T, \quad \forall i \in I
  \]

- Quality Constraint:
  \[
  x_i \cdot \hat{q}_i \geq x_i \cdot Q, \quad \forall i \in I
  \]
The Allocation Mechanism

Combinatorial Optimization Problem

maximize
social welfare

\[ \text{min} \sum_{i=1}^{n} \hat{C}_i x_i \]

All Tasks
Assigned

s.t. \[ \sum_{i=1}^{n} x_i = m \]

Time Constraint

\[ x_i \cdot \hat{t}_i \leq T, \quad \forall i \in I \]

Quality Constraint

\[ x_i \cdot \hat{q}_i \geq x_i \cdot Q, \quad \forall i \in I \]

Maximal number of tasks

\[ x_i \leq \hat{j}_i, \quad \forall i \in I \]
Maximize social welfare

\[
\min_{x_1, \ldots, x_n} \sum_{i=1}^{n} \hat{C}_i x_i
\]

Subject to

- All Tasks Assigned

\[
\sum_{i=1}^{n} x_i = m
\]

- Time Constraint

\[
x_i \cdot \hat{t}_i \leq T, \quad \forall i \in I
\]

- Quality Constraint

\[
x_i \cdot \hat{q}_i \geq x_i \cdot Q, \quad \forall i \in I
\]

- Maximal number of tasks

\[
x_i \leq \hat{j}_i, \quad \forall i \in I
\]

- No fractional tasks

\[
x_i \geq 0, \text{integer}, \quad \forall i \in I
\]
Allocation Mechanism

Combinatorial Optimization Problem

maximize \( \text{social welfare} \)

min \( \sum_{i=1}^{n} \hat{C}_i x_i \)

s.t. \( \sum_{i=1}^{n} x_i = m \)

All Tasks Assigned

Time Constraint
\( x_i \cdot \hat{t}_i \leq T, \quad \forall i \in I \)

Quality Constraint
\( x_i \cdot \hat{q}_i \geq x_i \cdot Q, \quad \forall i \in I \)

Maximal number of tasks
\( x_i \leq \hat{j}_i, \quad \forall i \in I \)

No fractional tasks
\( x_i \geq 0, \text{integer}, \quad \forall i \in I \)
Allocation and Pricing Mechanism

Algorithm

Require: $I, W, B, T, Q$

$\hat{\theta} = runProcurementAuction(I, QualificationTest, m)$

$x = allocationMechanism(\hat{\theta}, W, T, Q)$

if $x$ is feasible then

$p = paymentMechanism(\hat{\theta}, x)$

else

return no completion time feasible allocation found

endif

$costs = \sum_{i \in I} p_i \cdot x_i$

if $costs \leq B$ then

return $x, p$

else

return no budget-feasible solution found

endif
Allocation and Pricing Mechanism

Algorithm

Require: $I, W, B, T, Q$

$\hat{\theta} = runProcurementAuction(I, QualificationTest, m)$

$x = allocationMechanism(\hat{\theta}, W, T, Q)$

Vickrey-Clarke-Groves-Mechanism

costs = $\sum_{i \in I} p_i \cdot x_i$

if costs $\leq B$ then
    return x, p
else
    return no budget-feasible solution found
end if
Algorithm

**Require:** $I, W, B, T, Q$

$$\hat{\theta} = \text{runProcurementAuction}(I, \text{QualificationTest}, m)$$

$$x = \text{allocationMechanism}(\hat{\theta}, W, T, Q)$$

**if** $x$ is feasible **then**

$$p = \text{paymentMechanism}(\hat{\theta}, x)$$

**else**

**return** no completion time feasible allocation found

**end if**

$$\text{costs} = \sum_{i \in I} p_i \cdot x_i$$

**if** $\text{costs} \leq B$ **then**

**return** $x, p$

**else**

**return** no budget-feasible solution found

**end if**

---

Outline

1 Formal Model
2 Platform
3 Mechanism
4 Evaluation
5 Discussion
Experimental Set-up

• Simulation with 10’000 distinct trials

• **Baseline 1:** First-completed first-served allocation with ex-ante defined fixed prices

• **Baseline 2:** Optimal allocation (IP) with ex-ante defined fixed prices
### Results

#### Sanity Check

**Share of feasible (F) and non-feasible (NF) allocations found by each mechanism.**

<table>
<thead>
<tr>
<th>Budget</th>
<th>Mechanism</th>
<th>Baseline 1</th>
<th>Baseline 2</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>CMM</td>
<td>F</td>
<td>NF</td>
</tr>
<tr>
<td>100%</td>
<td></td>
<td>55%</td>
<td>16%</td>
</tr>
<tr>
<td></td>
<td></td>
<td>14%</td>
<td>15%</td>
</tr>
</tbody>
</table>
### Results

#### Sanity Check

Share of feasible (F) and non-feasible (NF) allocations found by each mechanism.

<table>
<thead>
<tr>
<th>Budget</th>
<th>CMM</th>
<th>Baseline 1</th>
<th>Baseline 2</th>
</tr>
</thead>
<tbody>
<tr>
<td>100%</td>
<td>F</td>
<td>55%</td>
<td>60%</td>
</tr>
<tr>
<td></td>
<td>NF</td>
<td>14%</td>
<td>23%</td>
</tr>
</tbody>
</table>

Price = \( \frac{\text{Budget}}{\text{Number of Tasks}} \)
share of feasible (F) and non-feasible (NF) allocations found by each mechanism.

<table>
<thead>
<tr>
<th>Budget</th>
<th>CMM</th>
<th>Baseline 1</th>
<th>Baseline 2</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>F</td>
<td>F</td>
<td>F</td>
</tr>
<tr>
<td>100%</td>
<td>55%</td>
<td>16%</td>
<td>60%</td>
</tr>
<tr>
<td></td>
<td>14%</td>
<td>15%</td>
<td>23%</td>
</tr>
<tr>
<td></td>
<td>15%</td>
<td>16%</td>
<td>6%</td>
</tr>
</tbody>
</table>
### Results

#### Sanity Check

<table>
<thead>
<tr>
<th>Budget</th>
<th>Mechanism</th>
<th>Baseline 1</th>
<th>Baseline 2</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>CMM</td>
<td>F</td>
<td>NF</td>
</tr>
<tr>
<td>100%</td>
<td>F</td>
<td>55%</td>
<td>16%</td>
</tr>
<tr>
<td></td>
<td>NF</td>
<td>14%</td>
<td>15%</td>
</tr>
<tr>
<td>10%</td>
<td>F</td>
<td>0%</td>
<td>71%</td>
</tr>
<tr>
<td></td>
<td>NF</td>
<td>1%</td>
<td>28%</td>
</tr>
</tbody>
</table>

Share of feasible (F) and non-feasible (NF) allocations found by each mechanism.
Share of feasible (F) and non-feasible (NF) allocations found by each mechanism.
Results

Number of Feasible Allocations

Table I: A comparison of the three mechanisms with respect to the percentage of cases where they found a feasible allocation.

<table>
<thead>
<tr>
<th>Mechanism</th>
<th>Feasible (F)</th>
<th>Non-Feasible (NF)</th>
</tr>
</thead>
<tbody>
<tr>
<td>CrowdManager</td>
<td>55%</td>
<td>16%</td>
</tr>
<tr>
<td>Baseline 1</td>
<td>28%</td>
<td>32%</td>
</tr>
<tr>
<td>Baseline 2</td>
<td>71%</td>
<td>29%</td>
</tr>
</tbody>
</table>

To get a better picture, consider now Figure 3. Here we graph all 10 different budget levels (i.e., price levels) used by the baseline mechanisms.

On the right side of Table 1, we see the situation for a very small budget of 10%, where CrowdManager clearly outperforms both mechanisms in terms of finding feasible allocations. As the budget increases, the performance of CrowdManager remains stable, while the performance of the baseline mechanisms decreases significantly.

This clarifies the overall picture: We see that for a budget of 100%, CrowdManager's mechanism outperforms both baseline mechanisms, with the number of feasible allocations being higher by about 28 percentage points. As the budget decreases, the advantage of CrowdManager over the baseline mechanisms becomes more pronounced, with CrowdManager always finding a strictly higher number of feasible allocations than the baseline mechanisms.

The results also show that when reducing the budget further, the number of feasible allocations found by the Baseline 1 mechanism is lower by about 15 percentage points and, hence, always strictly lower than that of CrowdManager. The Baseline 2 mechanism performs better at lower budget levels, but CrowdManager's advantage becomes more noticeable as the budget decreases.

When considering multiple budget levels (i.e., price levels) at once, plotting the number of feasible solutions found by all mechanisms allows for a more comprehensive understanding of their performance across different budget scenarios.
Results

Comparison of Average Costs

Consider Figure 4, where we plot the average cost for solving a work package in % of the budget by only comparing the cases where both mechanisms found a feasible solution. Naturally, as we decrease the available budget all the way down to 10%, we ultimately get to settings where the number of feasible allocations found by the baseline mechanism is very, very small (compare Figure 3). However, for those parameter settings where it does find an allocation, the total cost is obviously also very small (i.e., 10% of the overall budget). Thus, in exactly those cases it is expected that the total costs spent by the baseline mechanism is clearly able to increase the requestor's utility, which decreases as the budget is reduced. To get the full picture, and to determine the overall effect on the requestor's utility, we also need to consider the total amount of money the mechanisms will find allocations that are more cost effective than using a fixed-price mechanism. And, unless almost all of the available budget is spent, the total cost incurred by the mechanisms is spent for allocating their tasks to the agents. Here, we find that the VCG payment rule results in costs that lie at around 1/2 of the VCG payment rule only spends 30%. Even for those cases where the number of feasible allocations found by the baseline mechanisms use 100% of the budget, the VCG payment rule results in costs that lie at around 1/2 of the VCG payment rule. When the baseline mechanisms use 100% of the budget, the VCG payment rule is spent for allocating their tasks to the agents. Here, we find that the CrowdManager also finds the same or even more feasible allocations than the baseline mechanisms find feasible solutions.

Overall, these simulation results are very encouraging. They show that, for many parameter settings, the CrowdManager provides a principled way to determine the “right” price and does not rely on “guessing” the right fixed price.

Conclusion from the Simulation

The Requestor’s Average Utility

Utility = Budget - Costs
• Varying the number of tasks in to be allocated does not impact the execution time
• The execution time increases quickly as the number of workers in the retainer grows.

Results
Runtime Analysis

![Graphs showing execution time vs. number of tasks and workers](image-url)
1 Formal Model
2 Platform
3 Mechanism
4 Evaluation
5 Discussion
VCG vs. Requestor-Optimal

• CrowdManager’s mechanism leads to large cost savings in average, but does not directly minimize the requestor’s cost

• Myerson-style mechanisms do not easily generalize to multiple dimensions

• Benefit of using a Myerson mechanism is likely to be negligible in practice

[Bulow & Klemperer, 1996]
Future Work

• Modeling learning effects / economies of scales vs. weariness effects

• Distributional assumptions for the variability of human performance

• Evaluation in a real-world experiment
Conclusion

- finds more feasible allocations under the requestor’s budget, completion time, and quality constraints
- increases the requestor’s utility
- leads to more efficient allocation
- offers a principled way for dynamic price task
No longer a need for “guessing the right price”
CrowdManager
Combinatorial Allocation and Pricing of Crowdsourcing Tasks with Time Constraints

Patrick Minder, Sven Seuken, Abraham Bernstein, Mengia Zollinger