Eötvös Loránd Tudományegyetem

Természettudományi Kar

Csóka Endre

matematikus szak

Hatékony sztochasztikus mechanizmusok tervezése

Diplomamunka

témavezető:

Pintér Miklós Péter, egyetemi adjunktus

Budapesti Corvinus Egyetem, Matematika Tanszék

Budapest, 2008

Tartalomjegyzék

Rövid áttekintés a mechanizmustervezésről

A globalizáció és a növekvő munkamegosztás egyre fontosabbá teszi azt a kérdést, hogy az egyes, főként a saját önző érdekei szerint működő piaci szereplők, a továbbiakban játékosok, hogyan fognak viselkedni a rájuk vonatkozó játékszabályok függvényében, és hogy milyen szabályokkal tudjuk őket legjobban egy kívánt cél, például a társadalmi összérdek szolgálatába állítani. A kérdés fontosságát jól jelzi, hogy az 1996-os és a 2007-es közgazdasági Nobel-díjat is ebben a témában, vagyis a mechanizmustervezés terén elért eredményekért adták.

A mechanizmustervezés elmélete leginkább olyan piaci, üzleti, fiskális, politikai–választási játékszabályok kidolgozásával foglalkozik, amelyek alkalmazása révén előre meghatározott, optimális eredményt érhetünk el. Leginkább olyan szituációkat, üzleti tranzakciókat vizsgál, amelyek során a játékosok között egyenlőtlenül oszlanak meg az információk, s ennek eredményeként úgy osztják meg egymás között az eszközöket, az erőforrásokat vagy az egymásnak nyújtott szolgáltatásokat, ahogy az például a gazdaság egésze vagy a össztársadalmi jólét szempontjából nem feltétlen lenne ideális. A cél olyan szabályrendszer kialakítása, amelynek alkalmazása révén a játékosok a saját profitmaximalizáló stratégiájukkal összességében épp a kívánt célt érik el. A mechanizmustervezést ezért néha fordított játékelméletnek is nevezik, hiszen míg a játékelméletben adott szabályok mellett keressük a létrejövő eredményt, addig a mechanizmustervezésben adott eredményhez keresünk játékszabályokat.

Eredményem a két, Nobel-díjjal elismert témakör közül inkább az 1996-oshoz köthető. Ekkor többek között William Vickreynek a második áras aukciókról való azon eredményét ismerték el, melynek Edward Clarke és Theodore Groves által továbbfejlesztett változata egy általános sokszereplős játékhoz mutat olyan mechanizmust, amiben minden játékosnak érdemes őszintén elmondani saját információit, és ezáltal a játékosok összprofitja maximalizálható. Formálisabban, minden játékos rendelkezésére áll a lehetséges cselekvéseinek egy-egy A_i halmaza, és adott egy $u : A_1 \times ...A_n \to \mathbb{R}^n$ függvény, ami megmondja, hogy attól függően, hogy ki milyen cselekvést választ, pusztán e cselekvésegyüttesből ki mennyi profitra tehet szert. Lényeges, hogy A_i és u_i az i. játékos privát információja. A mechanizmus ekkor az, hogy mindenki elmondja saját információit, ebből meghatározzák az összérdek szerinti optimális stratégiát, végül pedig mindenkit annyi pénzzel kompenzálnak, hogy a saját profitja megegyezzen azzal, amennyit részvételével hozzátett az összprofithoz. Ekkor pedig teljesül az, hogy senki sem jár jobban, ha saját információja helyett bármi mást mond el a többieknek, feltéve, hogy a többiek őszinték voltak. Vagyis az őszinte stratégiaegyüttes Nash-egyensúly.

Az eredmény számos hiányossága közül az egyik legnagyobb, hogy feltételezi, hogy a játékosok előre meg tudják mondani, hogy hogyan alakulna a munkájuk. Ezért ez például egy valódi ütemezési problémához nem is használható. Az eredményemben egy olyan mechanizmust mutatok, ahol az egyes játékosok munkamenete sztochasztikus, ezért az optimálisan ütemezett munka is folyamatos egyeztetéseket igényel, mégis mindenkinek érdeke lesz őszintének lenni, például nem érdemes nehezebbnek beállítani a feladatot vagy optimistábban nyilatkozni a munka aktuális állásáról, és ezzel elérhető a maximális hatékonyság.

Efficient design of stochastic mechanisms

Endre Csóka

Eötvös Loránd University, Budapest, Hungary

Abstract

In this paper, we focus on the problem where different risk-neutral players should cooperate while each of them has an own stochastic working process as private information, and the efficient cooperation would require the players to make decisions depending also on this information of others. Scheduling problems are a typical and important example. We will use the tool of certifiable communication, namely we allow such contracts that specify payments depending also on the communication between the players. With this tool, we design a mechanism that implements a fully cooperative behaviour.

1 Introduction

The implementation of stochastic dynamic multi-agent games such as scheduling problems raises a question that do not occur in other models: what if the desired hidden decisions of the agents depend on some earlier hidden chance events of other agents? Or using the classical terminology (e.g. Mas-Colell[5]), what if Nature tells some private information to some agents not at the beginning but during the process, and we want to implement such strategy profile in which the actions of the agents depend on these informations of others, while neither these informations and these actions can directly be observed? For example, in a scheduling problem, the efficient coordination of different tasks would require the agents to be interested in telling the proper information about the actual state of his working process, and depending on these informations, to urge the agents exactly as much as it is worth according to the total interest. In this paper, we design a nontrivial stochastic generalization of the Clarke–Groves mechanism[4], which implements such revelation strategy profile of the agents that achieves the maximum possible expected total payoff of all players; and we design its "first price" version, too, which achieves the very same only in some circumstances such as in perfect competition, but which is resistant against coalitions.

In our model, there is a player called Principal(♀) and she can contract with some of the other players called agents(♂) to work with. Each agent faces a stochastic decision tree, as private information, consisting of decision points and chance points in which the agent and Nature chooses the path to continue, respectively. There is a result assigned to each leaf to describe what the agent would provide reaching this leaf. Beyond the monetary transfers by the contracts, Principal gets the utility corresponding to the set of the achieved results of the agents, and each agent pays the cost corresponding to the leaf. Communication is allowed thoughout the process and a contract can be made between Principal and each agent such that specifies a monetary transfer between them depending on both the achieved result of the agent and the communication between them. This way the decisions of an agent may be dependent of the earlier messages he gets from Principal, which may be dependent of the earlier messages Principal gets from other agents, which may be dependent of their chance events. So communication gives us the possibility of making the agents interested in making decisions depending on some earlier chance events of others.

We will use the following mechanism. Principal asks an offer for contract from each agent and she declares beforehand that she will use the strategy, including both the choice of the agents to contract with and her communication, that will maximize her minimum possible payoff. Surprisingly, this simple mechanism makes the agents interested in truthful strategies that make the mechanism efficient.

Let us first see the honest offer of a very simple stochastic agent, who could have completed his task either before an earlier event with p probability or only before a later event with 1-p probability. What he should offer is the following.

"Beyond a fixed fee, I agree that if you say any amount of money x before I start then if I complete later then I will pay you px, but if earlier then you have to pay me $(1-p)x$."

This way the agent gets $p \cdot (1-p)x - (1-p) \cdot px = 0$ money beyond the fix fee in expectedly, so he does not come off badly whichever x is chosen by Principal. On the other hand, if the earlier completion is better for Principal by equivalent to money m then Principal should choose $x = m$, because this way she can be certain to get the mean of the payoffs of the two results with the weights of their probabilities. So Principal considers this offer as valueable as an offer for surely achieving the "mean" of the results for the same cost. From the point of view of the agent, this is the same as if Principal "believes" these probabilities.

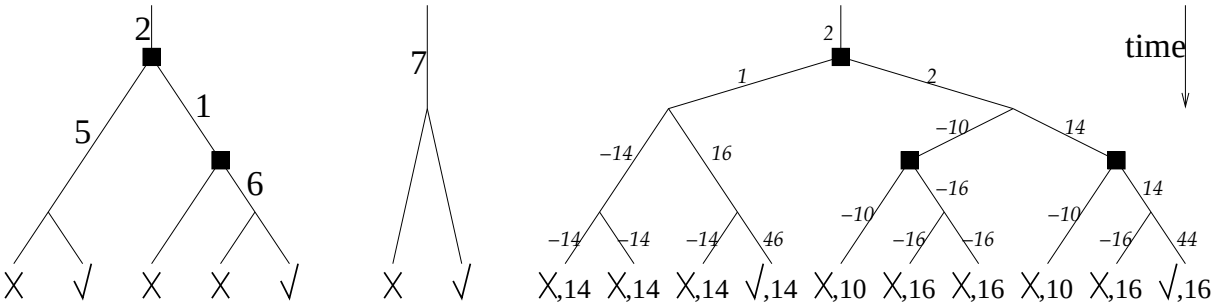

Figure 1

1.1 Example for the process

We consider a project that is very risky, but may gain huge utility. It consists of two tasks, one application per task must be accepted, and if both succeed in time then Principal gets a large sum of money, that is 60 here, but if either fails to do it then the success of the other task has no use. After getting all applications, Principal evaluates all pairs of them for the different tasks and accepts the pair producing her the most payoff. This example is only to show the evaluation of such a pair.

The first two trees describe the two applications in the following sense. The possible courses of the working process are shown by the paths from the root to a leaf. The solid squares show the decision points, at which the agent offers Principal to choose a branch to continue. It describes the possible decisions of the agent, for example choice of faster or cheaper shipment, number of employees, etc. The other branching points are the chance points. This describes, for example, the effect of an error, a failure of equipment, weather conditions, illness or simply faster or slower progress concerning the work. In this example, the path to continue is chosen randomly with 1/2 probabilities for each branch, and the agent agrees in taking any risk with 0 expected value. The numbers denote asked payments called costs in the case of the corresponding branch. At the bottom tick denotes the success and cross denotes the failure.

The first agent asks 2 units of money beyond his expenses. His task is either started at the beginning with a cost of 5 and probability of the success of 1/2, or he makes preparations for a cost of 1, and later if Principal wants then he can try to complete the task for a cost of 6 with probability 1/2. In the other application, the cost plus the desired expected payoff of the other task is 7 and the probability of the success is 1/2. It is important that *the timing of the events are represented by their heights.*

The third tree is a "product" of the two trees; that is, it describes their possible aggregate execution. We can construct it by following the applications by time, and

creating an appropriate branching if it occurs in one of them, and then continue this on both branches. For example, the path to the 5th leaf from the left describes the pair of the paths to the middle leaf in the first tree and the left leaf in the second tree. In details, the first agent chooses working by the second way, then the second agent fails to complete the task and then the first agent does not try to complete his task. At the bottom, tick denotes the success of both tasks, and cross denotes the failure of either, and the number shows the total payment asked by the agents. Let the value of an endstates be 60 if both tasks succeed and 0 otherwise, minus the total payments.

State means the state of the project at a point in time; it can be represented by a point in the graph of the third tree. We define the values of all states from the bottom to the top, and we denote them by italics. Values of states in the same edge are the same. The value before a decision point is the maximum of the values after. The value before each chance point is the mean of the values after.

As we will see, when Principal maximizes her minimum possible payoff then she surely gets the value of the starting state, because at the decision points she asks the decisions by which the value of the state remains the same, at the chance points she pays the signed difference between the values of the states after and before, and at the end she gets the value of the endstate. As for each agent, he gets the costs of his process, the extra payment he asked for and some random payment with expected value 0.

For the first agent this means that Principal asks him – a bit surprisingly – to work in the second way, that is to make only preparations, and then Principal either asks him to do nothing and he gets 3, or she ask him to try to complete the task, and he gets 9 and ± 30 for the risk, so he gets 39 if he succeeds, and he pays 21 if he fails. For the second agent this means that he gets 12 besides his costs if he succeeds (that is 19 in total), but Principal deducts 12 from his payment (that is he pays 5) in the case he fails.

(Note that these extremly high risks are the peculiarity more of this simple and risky example than of the mechanism. You can consider this project as an instance of many identical projects that should be made in parallel.)

2 General notions

For any symbol x, the definition of x_i-s will also define x as the vector of all meaningful x_i-s. x_{-j} means the vector of all except x_j.

The term of **actions** of a player refers to what the player effectively does, and

when he does them. The term of **information** of a player refers to what he knows and believes at a particular point in time. This point in time is always included in his information. His information in all points in time is called as his **information history**. The **strategy** set of a player is the set of all functions that assign some feasible action to each possible information. Therefore, a **game** can be defined by the players, their possible actions, information histories and payoffs. A **subgame** means the game from a point in time, given the earlier history. The **outcome $o_G(s)$** of a game G with a strategy profile s means the execution of the game, as a function of the other parameters of the game. The **preference** is a partial ordering on the outcomes or on the probability distributions of the outcomes. The preference of a player p is denoted by $\preceq_p$. A strategy profile is a Nash equilibrium if for any strategy s'_p of each player p, $o_G(s_{-p}, s'_p) \preceq_p o_G(s)$. To be **interested** in a strategy will mean that this gains him $\succeq_p$ outcome than any other strategy.

For the sake of transparency and gender neutrality, we use feminine or masculine pronouns depending on the gender (♀ or ♂) assigned to the player, and masculine pronouns for a not specified player.

3 The basic model

We model the problem as the following game G.

Players. There are a player C called **Principal**(♀) and some players $a_1, ...a_n$ called **agents**(♂). In subscripts, we use i instead of a_i.

Each agent has a stochastic **decision tree**, which is a rooted branching tree structure consisting of the following terms. The inner nodes of the tree are of two kinds: the **chance points** and the **decision points**. For each chance point, there is a probability assigned to each edge leaving it, with the sum of 1. The third kind of nodes are the leaves. Each of them has a **result** and a real **cost** assigned. There is a positive real point in time assigned to every node, and for each edge the time assigned to the parent is not later than to the child. For the sake of simplicity, we assume that all such points in time are different.

Working according to the decision tree of an agent means the following. The possible courses of his working process are the paths from the root to a leaf. In the case of a path, the result of the leaf is achieved at its time for the cost of the leaf. At a decision point, he can choose on which path to continue. At a chance point, the path is chosen randomly with the assigned probabilities, and the agent learns this choice. We call these choices as chance events. All chance events of all agents are independent. We denote the result and the cost of the reached leaf of such an

agent a_i who has been working by $\boldsymbol{r_i}$ and $\boldsymbol{c_i}$, respectively.

The actions of the players are described by the following process, in chronological order. For the sake of lucidity, this description includes the main events about getting information.

- Principal makes her strategy public. (Roughly speaking, she can define the rules.)
- Principal learns a **utility function** $\boldsymbol{u}$: {all possible sets of results} $\to \mathbb{R}$ and each agent learns his decision tree.

After these initial steps, each agent can send any costless certifiable time-stamped instant messages to Principal and vica versa.[1] Beyond this, the process of the game is the following.

- Until time 0, each agent tells to Principal whether he stays in the game.
- At time 0, Principal chooses some of those agents who stay in the game.
- Each chosen agent works according to his decision tree.
- Principal observes the results of the agents and pays some money to the chosen agents, which can also be negative.

The payoffs are determined as follows. Beyond the monetary transfers at the end, Principal gets $u(r)$, and each chosen agent a_i pays c_i. The payoffs of the not chosen agents are 0.

The informations of the players are defined only after the two initial steps. The **basic information** of an agent at any point in time consists of his decision tree, all messages he recieved earlier from Principal, whether Principal already allowed him to work and his chance events up to and including this time. The basic information of Principal at a point in time is her utility function u, the agents' decisions (if already made) about whether they stay in the game and the earlier messages she got. All we assume about the informations of the players are the followings.

- The information of each player includes his basic information.
- The chance event of each reached chance point at any time t has the probability distribution described in the decision tree, and given this description, conditionally independently of the informations of all other players until t and the information of this agent before t.

[1] To avoid some analytical problems, we only allow strategies by which the player surely sends finitely many messages.

Expectation $\boldsymbol{E}$ is taken only **over the chance events**. By default, the preference of each agent is to achieve higher expected payoff. (Namely, $o_G(s) \succeq_i o_G(s')$ iff with any feasible way of getting informations of all players depending on their earlier actions and not later chance events, the expected payoff of a_i is not smaller with the strategy profile s than with s'.)

3.1 Interpretation of the model

At the beginning, Principal designs the mechanism by declaring her strategy. Then she can negotiate with each agent, and they can agree on the terms (using her predefined protocol). If they do so then the agent can start to work.

Each agent can affect his work, for example by using more or less workers or be more hurry for some more fatigue – equivalent to some more cost. Furthermore, the agent gets feedbacks from his work, such as about unexpected failures, or simply faster or slower progress. The dynamics of them is described by his decision tree.

The players cannot completely observe the executions of other agents; for example, it can be happen that no other player can be sure that a later completion of the task of an agent was achieved by a fast and expensive work but with bad luck or by a slow and cheap work with average luck. We handle this aspect by the term of result.

The result can be interpreted so as the certifiable aspects of the execution of the task. So we assume that the certifiable aspects of the works of the agents determine the utility of the overall result for Principal. To put it the other way around, let us call some possible executions of a particular task *equivalent* if the utility is always the same with both executions, independently of the executions of others. Then the results are the equivalence classes of the executions, and the model requires the result to be certifiable.

In the model, the agents have seperate decision trees. However, in many cases like in scheduling problems, the work of an agent may restrain the actions of another one. For example, one cannot start building the roof of a house before someone completes the walls. To resolve this problem, we say that one could start building the roof before the walls are completed, but the result would be extremely bad.

We formalise this through the utility function u, namely, we consider u be $infty$ at all sets of executions that is impossible in reality. As u is defined not for sets of executions but for sets of results, this formalism implies the restricting and the restricted events to be certifiable. For the example, this means that each result of the agent building the walls must contain the completion time t_c, each result of

the agent building the roof must contain the starting time t_s, and if $t_s < t_c$ then $u = -\infty$.

To sum up, this model can be applied also in such cases when one's possible decisions can be restricted by some certifiable actions of other agents.

3.2 The goal

Definition 1. *We call the sum of the payoffs of all players the* ***payoff of the system****, and we denote it by* $\boldsymbol{p_s}$*. Clearly,* $p_s = u(r) - \sum c_i$*. The* ***preference of the system*** *is to achieve higher* $E(p_s)$*.*

Based on the game G, we define the following game $G^{(1)}$ describing the case when all players in G were the same. This game is to define the fully cooperative behaviour.

- There is only 1 *player*(♀).
- Her *actions* are choosing some of the decision trees and making the decisions during the working processes according to these trees.
- Her *information* is the utility function u, all decision trees and the not later chance events in all chosen trees.
- Her *payoff* is $p_s = u(r) - \sum c_i$.

Fixing u and the trees, p_s is a function of her strategy and the chance events, so $E(p_s)$ depends only on her strategy. As she can simulate any strategy profile in G, the maximum $E(p_s)$ on all strategies in $G^{(1)}$ is an upper bound of $E(p_s)$ in G. The similar statement holds with any subgame H of G, and we denote the maximum $E(p_s)$ in the corresponding one-player game $H^{(1)}$ by $\boldsymbol{f(H)}$ (as a function of u and the trees).

The upper bound of the information of an agent is called his **extended information**, which consists of the followings.

- In the subgame from the actual time, how the outcome depends on the chance events and on the strategies of the players;
- his chance event at the actual time;
- the probability distributions of the other chance events described in the decision trees.

So the information of an agent must be between his basic and extended information. Let his strategy set with basic or extended information mean his strategy set provided that he has basic or extended information history.

Definition 2. *With a fixed strategy of Principal, we call a strategy profile of the agents consisting of strategies with basic information as an* ***information-invariant Nash equilibrium****, if each agent is interested in keeping his strategy among his strategy set with extended information.*

Our main goal is to (weakly) information-invariantly Nash implement $f(G)$ as $E(p_s)$, namely to find a mechanism and a strategy profile by which

- $E(p_s) = f(G)$;
- the strategy profile is an information-invariant Nash equilibrium in the mechanism.

4 The mechanism

The **mechanism** means the subgame after Principal declares her strategy.

Definition 3. *An* ***application*** *of a_i is a function* $\boldsymbol{app_i} : \{$*all possible results*$\} \times \{$*all possible further communications between him and Principal*$\} \to \mathbb{R}$*, describing his asked payments in the different cases.*

We call the following subgame the **first price mechanism**. The process is the following.

1. Each agent a_i submits an application app_i.
2. Principal accepts or rejects each application.
3. Agents with accepted applications can work according to their decision trees, and each of them can communicate with Principal.

Let $Acc = \{i | app_i \text{ is accepted}\}$. At the end, the payoff $\boldsymbol{p}$ of a rejected agent is 0. If the communication between a_i and Principal at step 3 is denoted by $\boldsymbol{com_i}$, then

- $p(a_i) = app_i(r_i, com_i) - c_i$ if $i \in Acc$,
- $p(C) = u(r) - \sum\limits_{i \in Acc} app_i(r_i, com_i)$.

Let $com_{i\to C}$ and $com_{C\to i}$ denote the communication from a_i to Principal and from Principal to a_i, respectively. (Thus $com_{\to C} = (com_{i\to C}|i \in Acc)$ and $com_{C\to} = (com_{C\to i}|i \in Acc)$.) The above description still does not define Principal's choices of Acc and $com_{C\to}$. From now on, **Principal's strategy** means her strategy on these actions (which is also common knowledge). Denote her strategy for $com_{C\to}$ by s_c.

After step 1, $p(C)$ depends only on s_c, $com_{\to C}$ and r. **Principal chooses a strategy by which $\min_{com_{\to C}, r} p(C)$ is maximal**. **Maximin payoff** will refer to this value.

We assume that this rule determines a unique set of applications to accept, that there are no such ties.

4.1 Second price mechanism

Definition 4. *For any set S of applications, we define the **value from the applications**, denoted by **vf**(S), as Principal's maximin payoff if she recieves these applications. **Surplus value of an application** app_i means $\boldsymbol{v}_+(app_i) = vf(app) - vf(app_{-i})$.*

The second price mechanism[2] is the same as the first price one but Principal pays $v_+(app_i)$ more to each agent with application app_i.

So, denoting the payoffs in the second price mean by $\boldsymbol{p_2}$,

- $p_2(a_i) = app_i(r_i, com_i) + v_+(app_i) - c_i$,
- $p_2(C) = u(r) - \sum_{i=1}^{k}(app_i(r_i, com_i) + v_+(app_i))$.

By default, we consider the first price mechanism, and payoff refers to the first price payoff p.

5 Prime cost and fair strategies of agents

We define the **cost price application $\boldsymbol{pc}(a_i)$** of an agent a_i as the application which can be interpreted as follows. The agent shows his decision tree and defines the following communication protocol between him and Principal. At each decision point, Principal chooses a branch to continue and sends it to the agent. Before each chance point, Principal sends to the agent such real assignments to the branches that's mean weighted by the probabilities is 0. At each chance point, the agent chooses a branch to continue and sends it to Principal. At the end, Principal has to

[2]the name comes from the analogy to the second price sealed bid auctions of Vickey[7].

pay the cost of the leaf plus for each reached chance point, the money assigned to the chosen branch; and the agent has to deliver the result corresponding to the leaf in time. (If any player deviates from the protocol then, for example, he pays ∞.)

$\boldsymbol{fair}(a_i, x) = pc(a_i) + x$ is called a **fair application** for any $x \in \mathbb{R}$ called **profit**.

Fair strategy $\boldsymbol{F_x}$ of an agent a_i means submitting $fair(a_i, x)$, and in the case of acceptance, choosing the decisions corresponding to Principal's choice at each decision point, and sending the corresponding message at each chance point. Fair agent means an agent with fair strategy, and F_0 is called the **cost price strategy**. Clearly, fair strategies use basic imformation.

5.1 Evaluation of fair strategies

In this section, we describe more precisely the evaluation shown in the introductory example (1.1).

Consider each agent with fair application with profit x as an agent with x more cost in each leaf of his tree, and with cost price application.

We use the term of the **combined decision tree** of a set of agents. We can construct it by following the trees of the agents, and creating an appropriate branching in the combined tree if it occurs in one of the trees, and then we continue on all branches. Each result of the combined tree is equivalent to the set of the appropriate results.

Principal evaluates all subsets of *app*, and accepts all applications in the best set. Consider the evaluation of such a subset. We handle the set as a **combined application** meaning the offer for contract with all in the set. Notice that the combined application of cost price applications is the cost price application of the combined decision tree. (Combined applications can easily be written in the form of application, but we omit the details.)

A **state** of the combined decision tree means a point (not necessarily vertex) of the graph of the tree, like in the third tree in Figure 1.

Definition 5. *For any state T and expression X, let $X|\boldsymbol{T}$ denote X in the imagined case when the only application Principal accepts is the cost price application of the subtree from T. Let the **value of the state** be $\boldsymbol{v(T)} = \max\limits_{s_c|T} \min\limits_{(com_{\to C}|T),r} p(C)|T$.*

Notice that accepting this application is equivalent to accepting all cost price applications of the corresponding subtrees of the agents, and the value of the starting state is Principal's maximin payoff provided that she accepts this set of applications.

Theorem 1. *The values of the states can be calculated by backward recursion using the followings.*

Step 1. The value of an endstate is $u(r) - \sum c_i$

Step 2. Values of states in the same edge are the same.

For each inner node, denote the state just before, and the states just after the node by T *and* $T_1, ...T_n$*. For chance points, denote the probabilities by* $w_1, ...w_n$*, respectively.*

Step 3. For a decision point, $v(T) = \max_i v(T_i)$*.*

Step 4. For a chance point, $v(T) = \sum w_i v(T_i)$*.*

Furthermore, Principal gets the value of the starting state as a fix payoff.

Proof It is clear that these steps enable us to use backward recursion. The first two steps are right by definition.

Step 3. At the only message before the decision point, Principal should ask the corresponding agent for choosing a specified branch to continue. That is why,

$$v(T) = \max_{s_c|T} \min_{(com_{\to C}|T),r} p(C)|T = \max_i \max_{s_c|T_i} \min_{(com_{\to C}|T_i),r} p(C)|T_i = \max_i v(T_i).$$

Step 4. Let $x = \sum w_i v(T_i)$. At the only message before the chance point, Principal should send the assignment vector $t = (t_1, ...t_n)$ to the branches with $(\sum w_i t_i =) wt = 0$. Then, at the point in time of the chance point, the agent should reply a branch. That is why,

$$v(T) = \max_{s_c|T} \min_{(com_{\to C}|T),r} p(C)|T = \max_{t|wt=0} \min_i \max_{s_c|T_i} \min_{(com_{\to C}|T_i),r} ((p(C)|T_i) - t_i) = \max_{t|wt=0} \min_i (v(T_i) - t_i). \tag{1}$$

$t_i = v(T_i) - x$ satisfies

$$\sum w_i t_i = \sum w_i (v(T_i) - x) = (\sum w_i v(T_i)) - x = 0,$$

so using this t at (1), we get

$$v(T) \geq \min_i (v(T_i) - (v(T_i) - x)) = \min_i x = x. \tag{2}$$

On the other hand,

$$v(T) = \max_{t|wt=0} \min_i (v(T_i) - t_i) \leq \max_{t|wt=0} \sum w_i (v(T_i) - t_i) = \max_{t|wt=0} \sum w_i v(T_i) - \sum w_i t_i = x. \tag{3}$$

So from (2) and (3), we get $v(T) = x = \sum w_i v(T_i)$.

This recursion also shows that Principal gets the value of the starting state as a fix payoff. □

5.2 Efficiency of cost price or fair strategies

Lemma 2. *The expected payoff of an accepted agent a_i with fair strategy and accepted application $fair(a_i, x)$ is the profit x.*

Proof At each chance point, given its probability distribution described in the decision tree, Principal's assignments to the branches are conditionally independent of the chance event, so the expected value of the assignment to the chosen branch is 0. That is why $E(p(a_i))$ is independent of these assignments, so let us consider them as 0 at each branch. In this case, a_i gets from Principal his cost plus the profit x, that is why his payoff is x. To $E(p(a_i)) = x$. □

Proposition 3. *Consider the subgame G^- of either the first or second price mechanism, starting after the choice of Acc. In this game, if all agents use fair strategy then $E(p_s) = f(G^-)$ and $p(C)$ is fixed.*

Proof Let a_i use the strategy F_{x_i}. Let us denote the subgame starting from a state T by G_T. We can determine $f(G_T)$ by the same backward recursion as at Theorem 1. At each endstate, the payoffs of the players are fixed, so instead of Step *1.*, we have that for each endstate T,

$$f(G_T) = E(p_s)|T = p(C)|T + \sum p(a_i)|T = v(T) + \sum x_i.$$

Steps *2.*, *3.* and *4.* are valid with $f(G_T)$ instead of $v(T)$, too. And it is easy to check that using these steps, we get by induction that $f(G_T) = v(T) + \sum x_i$ holds for every T, including the starting state T_0. Thus,

$$f(G^-) = f(G_{T_0}) = v(T_0) + \sum x_i \leq p(C) + \sum E(p(a_i)) = E(p_s). \qquad (4)$$

On the other hand, $f(G^-) \geq E(p_s)$, so $f(G^-) = E(p_s)$, and equality holds at (4), thus $p(C)$ is fixed. □

Theorem 4. *If all agents use cost price strategy then $E(p_s) = f(G)$.*

Proof Because of the previous theorem, what we only have to show is Principal chooses the best set of agents according to the preference of the system. Whichever set is chosen by Principal, the expected payoff of each agent is 0, so $E(p_s) = p(C)$.

That is why, when Principal chooses such set of agents by which her minimum payoff becomes the largest possible, then she chooses the set by which $E(p_s)$ becomes the largest possible. □

6 Interest in cost price strategy in the second price mechanism

Theorem 5. *In the second price mechanism, cost price strategy profile is an information-invariant Nash equilibrium.*

Proof 1 The outline of the proof is as follows. We consider an agent which we denote by a_1 and assume that all other agents use cost price strategy. Then we will prove that $E(p_2(a_1)) \leq f(G) - vf(app_{-1})$, and equality holds if a_1 also uses cost price strategy. As the right side is independent of the strategy of a_1, this will prove the equilibrium.

Consider a mixed mechanism that is second price for a_1 and first price for the other agents, namely the payoffs of the players are the followings:

- $p^*(a_1) = app_1(r_1, com_1) + v_+(app_1) - c_1$, or 0 if rejected;
- $p^*(a_i) = app_i(r_i, com_i) - c_i$ for all $i \neq 1$, or 0 if rejected;
- $p^*(C) = u(\{r_i | i \in Acc\}) - \sum_{i \in Acc} app_i(r_i, com_i) - v_+(app_1)$.

Clearly, the payoff and the preference both of a_1 and of the system are the same here as in the second price mechanism. Lemma 2 shows that $p^*(a_i) = p(a_i) = 0$ for all $i \neq 1$. Theorem 4 shows that if a_1 uses cost price strategy then $E(p_s)$ is the largest possible.

$$p^*(C) = p(C) - v_+(app_1) \geq vf(app) - (vf(app) - vf(app_{-1})) = vf(app_{-1}),$$

and equality holds when a_1 is fair. Consequently,

$$E(p^*(a_1)) = E(p_s) - E(p^*(C)) - \sum_{i \neq 1} E(p^*(a_i)) = E(p_s) - E(p^*(C)) \leq f(G) - vf(app_{-1}),$$

and equality holds if a_1 also uses cost price strategy. □

Proof 2 We use the same solution concept. Using the equations

$$\min_{com \to C, r} (p_2(C)) = vf(app) - \sum v_+(app_i) \tag{5}$$

and if $app_i = pc(a_i)$ then $E(p_2(a_i)) = v_+(app_i)$, we get

$$E(p_s) - E(p_2(a_1)) = E(p_2(C)) + \sum_{i\neq 1} E(p_2(a_i)) \geq \min_{com\to C,r}(p_2(C)) + \sum_{i\neq 1} E(p_2(a_i)) \overset{(5)}{=}$$

$$vf(app) - \sum v_+(app_i) + \sum_{i\neq 1} E(p_2(a_i)) = vf(app) - v_+(app_1) = vf(app_{-1}),$$

so

$$E(p_2(a_1)) \leq f(G) - vf(app_{-1}).$$

□

Corollary 6. *The second price mechanism information-invariantly Nash implements $f(G)$ as $E(p_s)$.*

7 Interest in fair strategy in the first price mechanism

Definition 6. *The **value** v **of an application** is Principal's maximin payoff on her such strategies by which she accepts this application; that is*

$$v(app_i) = \max_{(Acc|i\in Acc),s_c} \min_{com\to C,r} p(C)$$

$v(app_i) \leq vf(app)$, with equality iff app_i is accepted. Furthermore, app_i is accepted iff $v(app_i) > vf(app_{-i})$. That is why $v_+(app_i) = \max(0, v(app_i) - vf(app_{-i}))$.

The value of a fair application is the value of the cost price application minus the profit. So for a particular agent, there is exactly one fair application with a given value.

Theorem 7. *For an arbitrary agent a_i and value x, if every other agent is fair then the fair strategy of a_i gains him the highest expected payoff among all those strategies that use an application of this value.*

Proof Consider again the profits of the other agents as constant costs, so their strategies are considered to be cost price strategies; it does not modify the preference of the system. As the acceptance depends on the value, we should consider only the case when the value is big enough to accept. Let us increase the cost of each leaf of the tree of a_i by as much as makes the value of his cost price application x. It decreases his payoff by a fix amount, so it does not modify his preference.

$$E(p_s - p(a_i)) = E(p(C)) + \sum_{j \neq i} E(p(a_j)) = E(p(C)) \geq \min_{com \to C, r}(p(C)) = v(app_i),$$

so $E(p_s) - v(app_i) \geq E(p(a_i))$. If a_i uses cost price strategy then equality holds and Theorem 4 shows that it makes $E(p_s)$ the largest possible. That is why, the best for a_i among such strategies is his cost price strategy. With the original costs, this means that the best for him is his fair strategy of the value. □

First we describe the result more intuitively, then we present the formal results in the subsections.

For a strategy s_i with application app_i, and for a real number x, let $s_i + x$ mean the same strategy as s_i but with application $app_i + x$, and let the emphstrategy form $Form(s_i) = \{s_i + x | x \in \mathbb{R}\}$. Let $F = \{F_x | x \in \mathbb{R}\}$ called the fair strategy form.

Let us fix everything in the game except the chance events and the strategy of a_i. Let $p_i(s_i)$ mean $E(p(a_i))$ if he uses the strategy s_i. If both applications app_i and $app_i + x$ would be accepted then $p_i(s_i + x) + v(app_i + x) = (p_i(s_i) + x) + (v(app_i) - x) = p_i(s_i) + v(app_i)$,[3] so this sum depends only on the strategy form. We call this sum as the *value of the strategy form* $v(Form(s_i))$.

In each strategy form $Form$ there exists an only strategy s_i, using application app_i, for which $app_i - x$ would be accepted if $x > 0$ and rejected if $x < 0$. Then $p_i(s_i - x) = \{0 \text{ if } x < 0; v(Form(s_i)) - x \text{ if } x > 0\}$.

$p_i(s_i) \leq v(Form(s_i))$, so his strategy form with the greatest value gains to a_i the highest potential for his expected payoff. Another question is expectedly how much he could exploit this potential, in the Bayesian mean.

Theorem 7 implies that the fair strategy form has the greatest value. Moreover, $p_i(F_x) = \{x \text{ if } v(F) > x; \text{ and } 0 \text{ otherwise}\}$, which is a quite efficient way for the exploitation of this potential.

7.1 Interest in cost price strategy with perfect competition

Principal accepts the applications with the greatest values. This justifies the following definition.

Definition 7. *We define the* ***perfect competition*** *as the mechanism with the following preferences of the agents. Each agent prefers the case when his expected payoff would be nonnegative if Principal chose her maximin strategy on her such*

[3] In fact, we assume here that the actions of the other agents are independent of the choice of x.

strategies by which she accepts the application of this agent. Of those, he prefers submitting an application with the greater value.

Theorem 8. *In first price mechanism with perfect competition, cost price strategy of all agents is an information-invariant Nash equilibrium.*

Proof Assume that every agent except a_i uses cost price strategy. What we have to prove is a_i is interested in using cost price strategy, as well. If $p(a_i) \geq 0$ then

$$v(app_i) = \min_{com_{\to C},r}(p(C)) \leq E(p(C)) = E(p_s) - \sum_{j \neq i} p(j) = E(p_s) - p(a_i) \leq f(G),$$

and equality holds if a_i also uses cost price strategy. So this inequality gives an upper bound for his preference, which can always be achieved with cost price strategy. □

Corollary 9. *The first price strategy information-invariantly Nash implements $f(G)$ as $E(p_s)$ with perfect competition.*

7.2 First price mechanism with imperfect competition

We use some imprecise descriptions when it is unambiguous and the precise way would be too broad.

We assume here that the information of each agent a_i contains a feasible probability distribution on everything in the game including the strategies of others, and we use probability P_i and expected value E_i also on this probability distribution.

Definition 8. *Let the **signed surplus value of an application** app_i be $v_{\pm}(app_i) = v(app_i) - vf(app_{-i})$.*

Clearly, $v_+(app_i) = \max(v_{\pm}(app_i), 0)$ and $v_{\pm}(app_i) > 0$ iff app_i is accepted.

With a fixed agent a_i and application app_i, let (value) $\boldsymbol{V} = v_{\pm}(pc(a_i))$, (difference) $\boldsymbol{D} = V - v_{\pm}(app_i) = v(pc(a_i)) - v(app_i)$ and $\boldsymbol{e} = E_i(D|D < V)$.

In practice, V and D are almost independent and both have "natural" distributions, so $P_i(e < V) = P(E_i(D|D < V) < V)$ is usually not smaller than $P_i(D < V)$. This observation shows the importance of the following theorem.

Theorem 10. *If $P_i(E_i(D|D < V) < V) \geq P_i(D < V)$ holds for any i and app_i, and this is common knowledge among the agents then a fair strategy profile is a Nash equilibrium.*

Proof Denote the strategy of a_i by s_i using application app_i and let $g(s_i) = E_i(p(a_i)|s_i)$ and x be the number by which $g(F_x)$ is the largest possible. So if $g(F_x) \geq g(s_i)$ for all s_i then a_i is interested in using fair strategy.

We prove that if all but an agent a_i use fair strategy then a_i is interested in using a fair strategy. Theoretically, let us allow only for a_i to submit an application $fair(app_i)$ in which he submits the fair application with the same value as which $v(app_i)$ would be if he submitted app_i instead. Denote the fair strategy but with application $fair(app_i)$ by $\boldsymbol{F(app_i)}$.

app_i is accepted iff $v_{\pm}(app_i) > 0$, or equivalently, $D < V$. From

$$g(s_i) = P_i(i \in Acc) \cdot E_i(p(a_i)|s_i, i \in Acc),$$

we get $g(F_e) = P_i(e < V)e$, and $g(F(app_i)) = P_i(D < V)e$, whence we can simply get that

$$g(F_x) - g(s_i) = (P_i(e < V) - P_i(D < V))e + (g(F(app_i)) - g(s_i)) + (g(F_x) - g(F_e))$$

- If $e \leq 0$ then $g(s_i) \leq 0 = g(F_0) \leq g(F_x)$ so s_i cannot be a better strategy. That is why assume that $e > 0$. In this case, $(P_i(e < V) - P_i(D < V))e \geq 0$. If s_i is fair then D is constant, so both sides are the same.
- Proposition 3 implies that $g(F(app_i)) - g(s_i) \geq 0$.
- $g(F_x) - g(F_e) \geq 0$ by the definition of x.

To sum up, $g(F_x) - g(s_i) \geq 0$, which proves the equilibrium. □

7.3 Coalitions

In this section, we consider the case when we have disjoint sets of agents, called coalitions, and each agent in a coalition prefers the higher total expected payoff of all in his coalition.

Submitting more applications by the same agent is equivalent to submitting one application in which the agent shows all and offers Principal to choose some of them. That is why if a coalition played as one agent, called consortium, with their combined decision tree, their overall information and the default preference, then this consortium would be able to simulate the case of playing so as different agents in a coalition. So if allowed, each coalition would better forming a consortium, by which we get the original game with this new set of agents.

As a consequence, if the competition remains perfect with this new set of players then the mechanism remains efficient.

7.4 Reliance in Principal

Assume that Principal knows that all agents use fair strategy. Then, with the corresponding modifications, she can consider them as agents with cost price strategies. In this case the expected payoffs of all agents are 0. That is why $E(p(C)) = E(p_s) \leq f(G)$, and Theorem 4 shows that equality holds if Principal uses her declared strategy. To sum up, Principal is interested in choosing her declared strategy. (Even if Principal is risk-averse because she gets $f(G)$ as a fix payoff.)

8 Conclusion

In the second price mechanism, the revelation strategy profile is an information-invariant Nash equilibrium, and this way the expected payoff of the system is the largest possible, including the possibilities of making one's decision depending on the earlier chance events of others.

In the first price mechanism, with some assumption (or approximation), there is a Nash equilibrium consisting of the same strategies but with asking a constant more money – his expected payoff in the case of acceptance – in each application.

The *disadvantage* of the first price mechanism over the second price one is that it requires a competition on each task, and it is fully efficient only with perfect competition.

The *advantages* of the first price mechanisms are

1. forming cartels of agents does not worsen the process as long as it does not decrease the competition, while in the second price mechanism, if two agents can submit such applications that are useless without each other then they may get as much payoff as they want (see 10.1);
2. the agents need (almost) no reliance in Principal.

9 Extensions and consequences of the model

9.1 When Principal also has a decision tree

Consider the extension of the model when Principal also has a decision tree, and the utility depends on her result, too. Consider here this Extended Principal(♀) as two players, one is considered as an indispensable agent with the decision tree who declares that he uses cost price strategy, and the other is Principal in the original meaning with her strategy of the first or second price mechanism. In the second

price mechanism, we omit paying the surplus value of the application of this fictive agent. Then the expected payoff of Extended Principal and Principal are the same, and all statements hold with this new set of players, that is why this mechanism in the extended model is the same good as the original mechanism in the original model.

Using this strategy of Extended Principal is equivalent to choosing the strategy by which her minimum expected payoff only on her own chance events is the highest possible.

9.2 When agents also have utility functions

Consider the extension of the model when some agent a_i has a utility function u_i which is 0 if a_i is rejected, and may depend on the work of others if a_i is accepted, that is $p(a_i) = a_i(r_i, com_i) + u_i(o) - c_i$. Let us allow making such applications and contracts that specify the payment depending also on the results of other contracted agents. Then we define cost price application by the same way, but decreased by u_i. This way, with some corresponding changes, all that we have shown above remains true.

9.3 Modifications during the process

In practice, the decision trees can be extremely difficult, that is why submitting the precise fair applications is not expectable. Therefore, they can present it only in a simplified, approximating way. Generally, such inaccuracies do not significantly worsen the optimality, nevertheless, this loss can be much more reduced by the following observation.

Assume that someone whose application has been accepted can refine his decision tree during the process. It would be beneficial to allow him to carry out such modifications. The question is: on what conditions?

The answer is for us to allow modifications of applications, if the agent pays the difference between the values of the actual states with the original and the new applications. From another point of view, considering the possible applications with the restriction of the earlier communication, an agent can exchange his application to another one with the same value in the restricted mean. As it is shown at Theorem 7, exchanging to his true fair application is in his interest, and such modifications gains him as much more expected payoff as to the system.

Equivalently, each possible modification can be handled so as a chance point in the original application with 1 and 0 probabilities for continuing with the original

and the modified application, respectively. (Or more precisely, as the limit of the cases when these probabilities tend to 1 and 0.) Because at such chance point, Principal assigns 0 to the branch of not modifying and the difference between the values of the states after and before, to the modification.

It may happen that in the beginning it is too costly for some agent to explore the many improbable branches of his decision tree, especially if he does not yet know whether his application will be accepted; but later however, it would be worth exploring better the ones that became probable. This kind of in-process modifications is what we want to make possible. We show that the interest of each agent in better scheduling of these modifications is about the same as the interest of the system in it.

The expected payoff of an agent with an accepted fair application is fixed and for a nearly fair agent, the small modifications of the other applications have negligible effect. As the modifications of each agent have no influence on Principal's payoff and only this negligible influence on the expected payoff of other agents, the change of the expected payoff of the system is essentially the same as the change of the expected payoff of this agent. This confirms the above statement.

On the other hand, it is clear that if the utility function alters somewhat then everything can be rescheduled according to the new goals. Moreover, Principal is interested in describing her utility function in the same schedule as which is the best according to the preference of the system.

10 Further theoretical observations

10.1 Coalitions in the second price mechanism

In the second price mechanism, consider the case when two agents can submit such applications that are useless without each other. Assume that their cost price applications would be accepted. If either of them decreased the cost of his application by x then the surplus value of both application would increase by x, so totally they get $2x$ compensation, by which they get x more total payoff. Using this trick, these players can get as much payoff as they want.

10.2 Simplifications and the case with no parallel tasks

The messages Principal sends depend only on the earlier messages she got. That is why if an agent a_i is sure that Principal recieves no message from anyone else in the

time interval $I = [t_1, t_2]$ then, without decreasing the value of his application, a_i can ask Principal (in the application) to send at t_1 that what messages she would send during I depending on the messages she would have got from a_i before. Similarly, if Principal surely does not send any message to anyone else during I then the agent can send all his messages only until t_2, and during I, and Principal should send to a_i that which messages he would send depending on the messages he would have got before.

As an application, consider the following project. It consists of two tasks, and the second task can only be started after the first one accomplished. The result of each agent for the first task (called first agent) consists of his completion time C_1. The result of each second agent consists of his starting time S_2 and the time C_2 he completes; and his decision tree starts with doing nothing until an optional point in time that is S_2, and then he can start his work. The utility function is of the form $f(C_2)$ for some decreasing function $f : \{\text{time}\} \to \{\text{money}\}$ if $C_1 \geq S_2$, and $-\infty$ otherwise. In this case, Principal always communicates only with the agent who is just working at the time. So using the above observation, we can make simplified applications of the following form with the same values as of the fair applications.

In short, the first agents tell that for how much money would they complete the first task depending on the penalty, and the applied penalty for the chosen second agent is the loss form the delayed completion, and the penalty for the first agent is how much more the second agent asks if he can start later. Principal chooses the pair that gains her the most payoff.

Formally, the form of the application of the first agents is "We ask $h(C_1) - g_1(h)$ money for any $h : \{\text{time}\} \to \{\text{money}\}$ that is chosen by Principal at the beginning", and for the second agents this is "We ask $f(C_2) - g_2(S_2)$ money if we can start our work at S_2 and we complete it at C_2". $h(C_1)$ and $f(C_2)$ describe the penalties here. In the simplified fair applications, g_1 and g_2 are chosen in such a way that make their expected payoff independent of the arguments, if the agents use their best strategies afterwards.

If all applications are so then Principal chooses a pair for which $g_1(g_2)$ is the greatest. Then she chooses $h = g_2$ for the first agent, and this way Principal gets $f(C_2) - (g_2(C_2) - g_1(g_2)) - (f(C_2) - g_2(S_2)) = g_1(g_2)$ payoff.

If a first agent has no choice in his decision tree, that is his completion time C_1 is a simple random value, then he should choose $g_1(h) = E(h(C_1)) - c$, where c is the cost plus the profit.

10.3 Controlling and controlled players

For an example, consider a task of building a unit of railroad. An agent a_1 can make this task for a cost of 100, but with 1% probability of failure, which would cause a huge loss $10,000$. Another agent a_2 could inspect and in the case of failure, correct the work of a_1 under the following conditions. The inspection costs 1. If the task was correct then he does nothing else. If not, he detects and correct the failure with 99% probability for a further cost 100, but he does not detect, so he does nothing with 1% probability. If both of them use cost price strategy and they are the accepted agents for the task then the mechanism works in the following way.

At the end, a_i gets 101.99 but pays 199 (totally he pays 97.01) compensation if he fails. a_2 gets 1 if he correctly finds the task to be correct, he gets 200 if the task was wrong but he corrects it, but he pays 9800 if he misses correcting it.

Of course, with fair applications each of them gets his profit more payment. It can be checked that the expected payoff of each agent is his profit independently of the behaviour of the others, and Principal's payoff is fixed.

10.4 Omitting Principal's assignments

The fair agents make no use of Principal's assignments at the chance points. Let us investigate what if we skipped these messages from the mechanism. In this case, the payment to each agent a_i no longer depends only on r_i and com_i but it also depends on $com_{\to C}$. That is why it would require from the agents much more reliance in Principal. But everything else we have shown remained true without the use of these messages.

10.5 A consequence for 2-player cooperation

Consider the case when a player a simply wants to make an offer for cooperation with another player c. We can consider a as an extended agent as in Section 9.2, c as an extended principal as in Section 9.1 and the offer of a as an application. Then c should accept or reject the application depending only on her expected payoff if she accepted it. By definition, this payoff equals the value of the application, and Theorem 7 shows that the fair application of all applications with the same value gains to a the highest expected payoff in the case of acceptance.

11 Observations for application

11.1 Necessity of being informed about the own process

We assumed that none of the chosen players knew better anything about any chance event of any other chosen agent. We show here an example that fails this requirement and it makes the mechanism wrong. Consider two agents a and b that will surely be accepted. Assume that a believes the probability of an unfavourable event in his work to be 50 %, but another one, called B knows that the probability is 60%, he knows the estimation of a and he also knows that at a particular decision point of him, he will be asked for the decision corresponding to this chance event. It can be checked that if the application of a is fair then if b increases his costs in his application of the more probable case by an amount of money and decrease the costs in the other case by the same amount then the value of his application remains the same but this way, he bets $1:1$ with b on an event of 60% probability.

In order to restrain such losses, a could rightfully say that larger bet can only be increased on worse conditions. Submitting reasonable application with concave valuability function makes something similar, that is another reason to use this.

11.2 Risk-averse agents

Assume that an agent a_i has a strictly monotone **valuability** funcion $g : \mathbb{R} \to \mathbb{R}$ and he wants to maximize $E(g(p(a_i)))$. We have seen safety use of the case when g is concave in Section 11.1.

Definition 9. *We define an application* ***reasonable*** *as the same as the fair application with the only difference that at the end, Principal pays*

$$g^{-1}(g(cost\ of\ the\ leaf) + \sum_{chance\ event} (assigned\ value\ to\ the\ chosen\ branch)).$$

By a reasonable application, in the case of acceptance, the expected valuability of the utility of the agent is independent of Principal's choices. If all applications are reasonable then Principal's payoff remains fixed. If the agent is risk-neutral then the reasonable application is fair. These are some reasons why reasonable applications work "quite good". We do not state that it is optimal in any sense, but a reasonable application may be better than a fair application in the risk-averse case.

We note that if $g(x) = a - b \cdot e^{-\lambda x}$ then the evaluation works in almost the same way as with fair applications.

11.3 Agents with limited funds

This section is only a suggestion for the cases with such agents, and it is not optimal in any sense.

Our mechanism requires each agent to be able to pay so much money as the maximum possible damage he could have caused. But in many cases, there may be a plenty of agents who cannot satisfy this requirement. However, accepting such agent a may be a good decision, if a is reliable to some degree.

To solve this problem, a should find someone who has enough funds, and who takes the responsibility, for example for an appropriate fee. If the agent is reliable to some degree then he should be able to find such insurer player b. (It can be even Principal, but considered as another player.) This method may also be used when a has enough funds, but he is very risk-averse.

Here, a and b work similarly as a controlled and a controlling parties in Section 10.3. The difference is that b does not work here, and he knows the probability distribution of the result of a not from his own decision tree but from his knowledge about the reliability of a. This shows that the role of b can be combined with his role in Section 10.3.

11.4 Risk level

If all agents are fair then at each chance point, the money assigned to each branch is the difference between the expected payoffs of the system after and before. That is why, in many cases, the risks of the agents are acceptable. For example, in an ordinary case, being late can cause at most as much penalty as much loss could be caused by this much more late in the project.

11.5 Communication

The model requires the communication to be certifiable. This can simply be made using a cryptographic communication protocol.

11.6 Applications in genetic programming

This second price mechanism may also be useful in genetic programming, when we want to find an efficient algorithm for such a problem that can be distributed into slightly dependent subproblems, while the subprograms for these subproblems should cooperate, and their overall achievement is what we can evaluate. In this case we should experiment such subprograms parallelly that also submits applications,

and we simulate here a competitive market with a Principal that uses our second price mechanism.

If the form of all possible cost price applications for each subproblem is simple enough then we may need to experiment less parameters in each subproblems than in the original problem, and that is why this method converges faster.

12 Acknowledgements

Special thanks to **Miklós Pintér** and thanks to Mihály Bárász, Katalin Friedl and many other people.

References

[1] *Bergantinos, G, E Sánchez: How to distribute costs associated with a delayed project.*
Annals of Operations Research, Volume 109, Numbers 1-4 (January, 2002) pp. 159-174

[2] *Brânzei, Ferrari, Fragnelli, Tijs: Two Approaches to the Problem of Sharing Delay Costs in Joint Projects*
Annals of Operations Research, Volume 109, Numbers 1-4 (January, 2002) pp. 359-374

[3] *Estévez-Fernández, Arantza, Peter Borm, Herbert Hamers: Project games*
International Journal of Game Theory, Volume 36, Number 2 (October, 2007), pp. 149-176

[4] *Groves, Theodore: Incentives in teams*
Econometrica, Vol. 41, No. 4 (Jul., 1973), pp. 617-631

[5] *Mas-Colell, Andreu, Michael Dennis Whinston, Jerry R. Green: Microeconomic Theory*
Oxford University Press, 1995

[6] *Rothkopf, Michael H., Thomas J. Teisberg, Edward P. Kahn: Why Are Vickrey Auctions Rare?*
The Journal of Political Economy, Vol. 98, No. 1 (Feb., 1990), pp. 94-109

[7] *Vickrey, William: Counterspeculation, Auctions, and Competitive Sealed Tenders*

The Journal of Finance, Vol. 16, No. 1. (Mar., 1961), pp. 8-37.