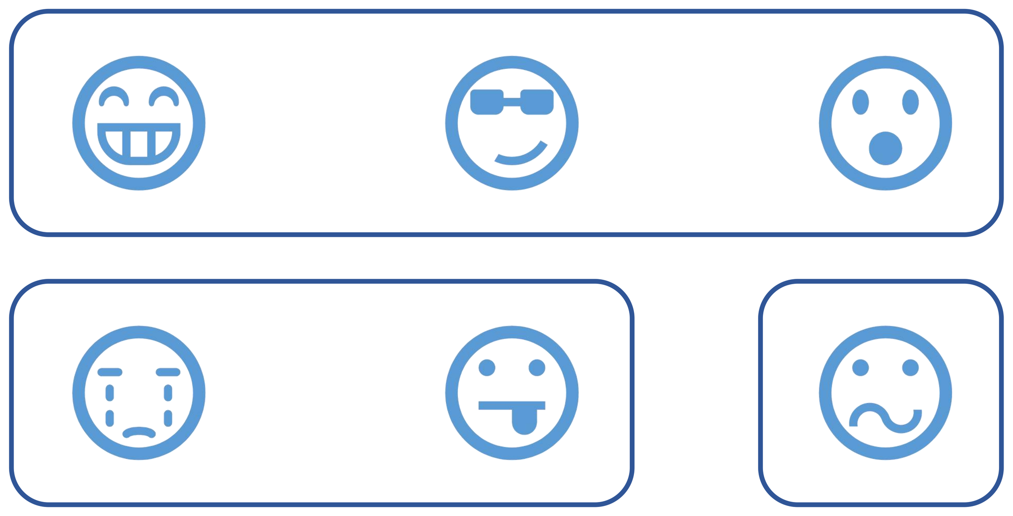
Learning Coalition Structures with Games

Yixuan Even Xu — Tsinghua
Chun Kai Ling — Columbia
Fei Fang — Carnegie Mellon

Coalition Structure Learning (CSL)

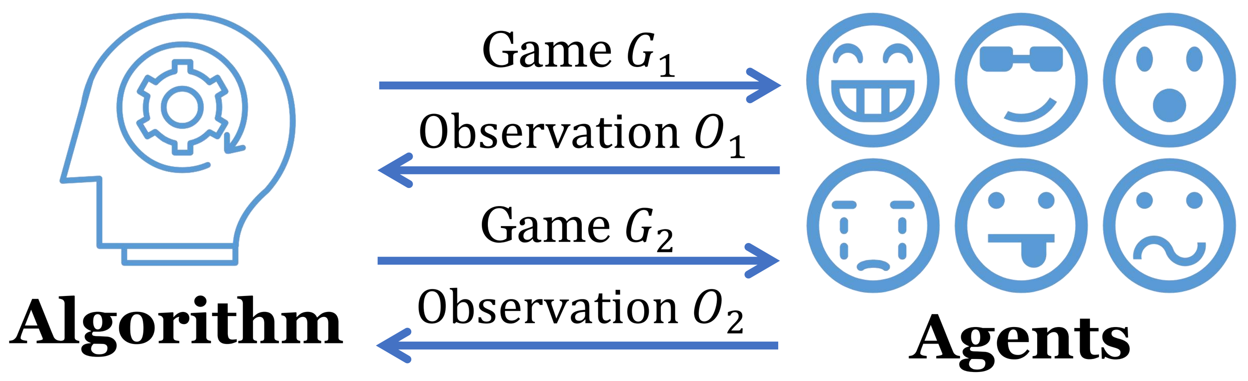


Coalition: A nonempty subset of the agents, in which the agents **coordinate their actions** and **have common interests**.

Coalition Structure: A set partition of the agents $\{1, 2, \cdots, n\}$

Behavior Model in a Game: Each coalition **act as a joint player** whose actual utility equals the **total utilities of its members**

Coalition Structure Learning (CSL): Recover the unknown coalition structure by observing interactions in designed games



*What kind of **games** can the algorithm design?*

*What **observation** does the algorithm obtain?*

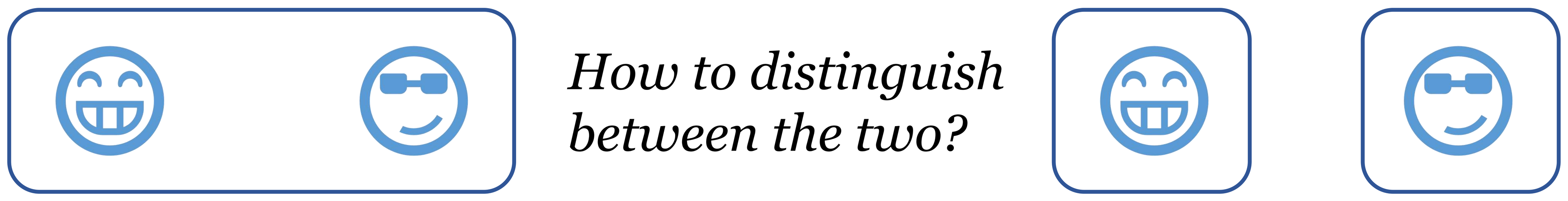
Single-Bit Observation Oracle: The algorithm queries a game G and a strategy profile Σ, the agents answer whether Σ is an **NE** in G

Easy to compute for the agents, **one bit of information** per query

Theorem 3.1: Any algorithm for CSL must interact **at least** $n\log_2 n - O(n\log_2\log_2 n)$ rounds with the agents

Types of Games: Normal form games, **congestion games**, **graphical games**, **auctions**. We study **all** four settings in this paper, and show **asymptotically optimal algorithms** for all of them.

Solving CSL with Normal Form Games



How to distinguish between the two?

Normal Form Gadgets: A normal form game where a specific pair of agents (x, y) play the **Prisoner's Dilemma**, and other agents only have one action that has no effect

	C_y	D_y
C_x	(3, 3)	(0, 5)
D_x	(5, 0)	(1, 1)

Lemma 3.1: (D_x, D_y) is an **Nash Equilibrium** if and only if x and y are **not in the same coalition**

Product of Normal Form Gadgets: Running several normal form gadgets simultaneously as **a single normal form game**

Agents **individually act** in each gadget

Agent's utility equals the **sum of that agent's utility** in each gadget

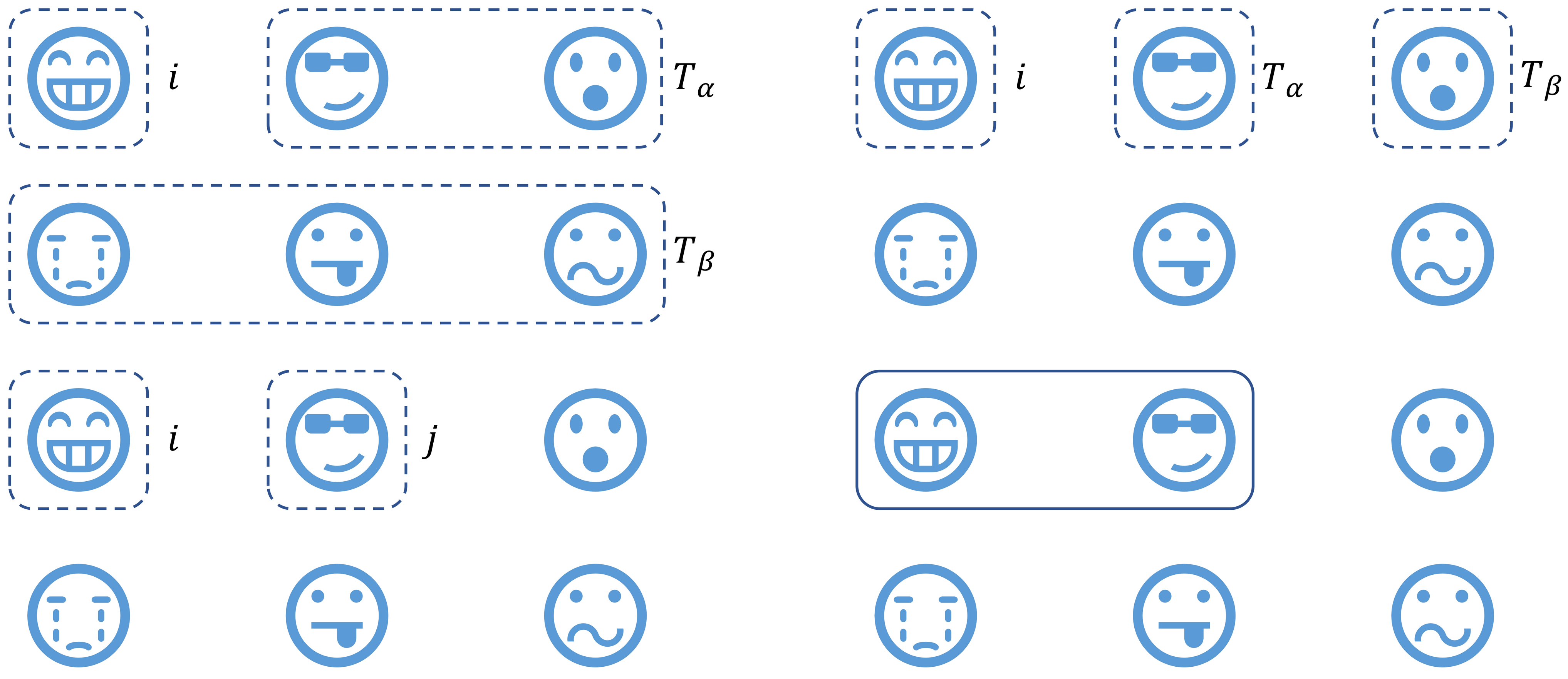
Lemma 3.2: Always defect is a **Nash Equilibrium** iff the chosen pair are **not in the same coalition in each gadget**

Our Algorithm: Iterative Grouping (IG)

Determine each agent's coalition one by one

For agent i, let all others play **normal form gadgets** with i

- If always defect is an NE, then agent i has **no other teammates**
- Otherwise, we know that **someone is in the same coalition** with i
- Run a **binary search** to locate one teammate j of i
  - **Merge** i and j as one joint player
  - Proceed iteratively until i's coalition is finalized



Theorem 3.2: IG solves CSL with $n\log_2 n + 3n$ rounds

*IG is **optimal** up to low order terms*

Extension: Solving CSL with Auctions

AuctionCSL: The algorithm can only design auctions

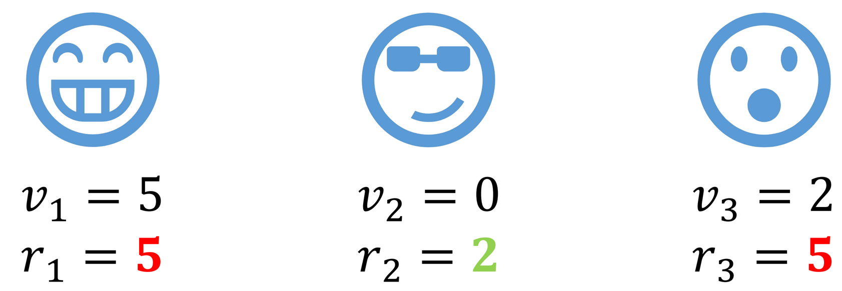
Format: Second-price auctions with personalized reserves

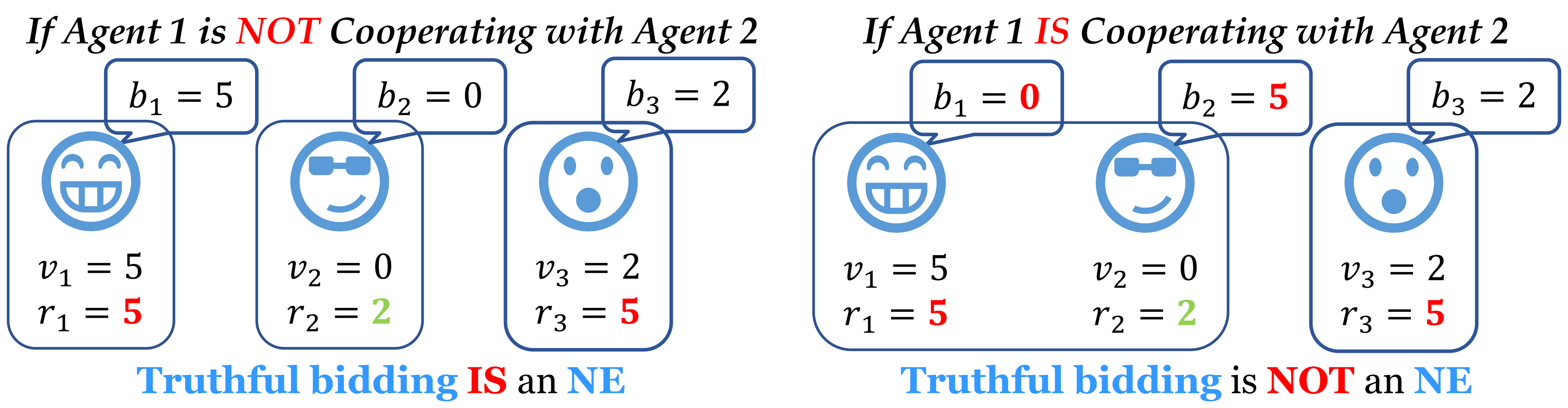
- Each agent i has a **valuation** v_i and a **reserve price** r_i
- The highest bidder wins, with $price = max\{second\ bid, reserve\ price\}$

To better simulate the practice, we further restrict the algorithm

- The algorithm can only design the **reserve prices**
- The **valuations** are random each query, but the algorithm sees them

***Auction Gadgets:** How to tell if there is cooperation between one specific agent and a group?*





AuctionIG: Our algorithm built upon auction gadgets

Theorem 4.1: In expectation, AuctionIG solves AuctionCSL with $(4.16 + o(1))n\log_2 n$ rounds, i.e., *AuctionIG is **optimal** asymptotically*