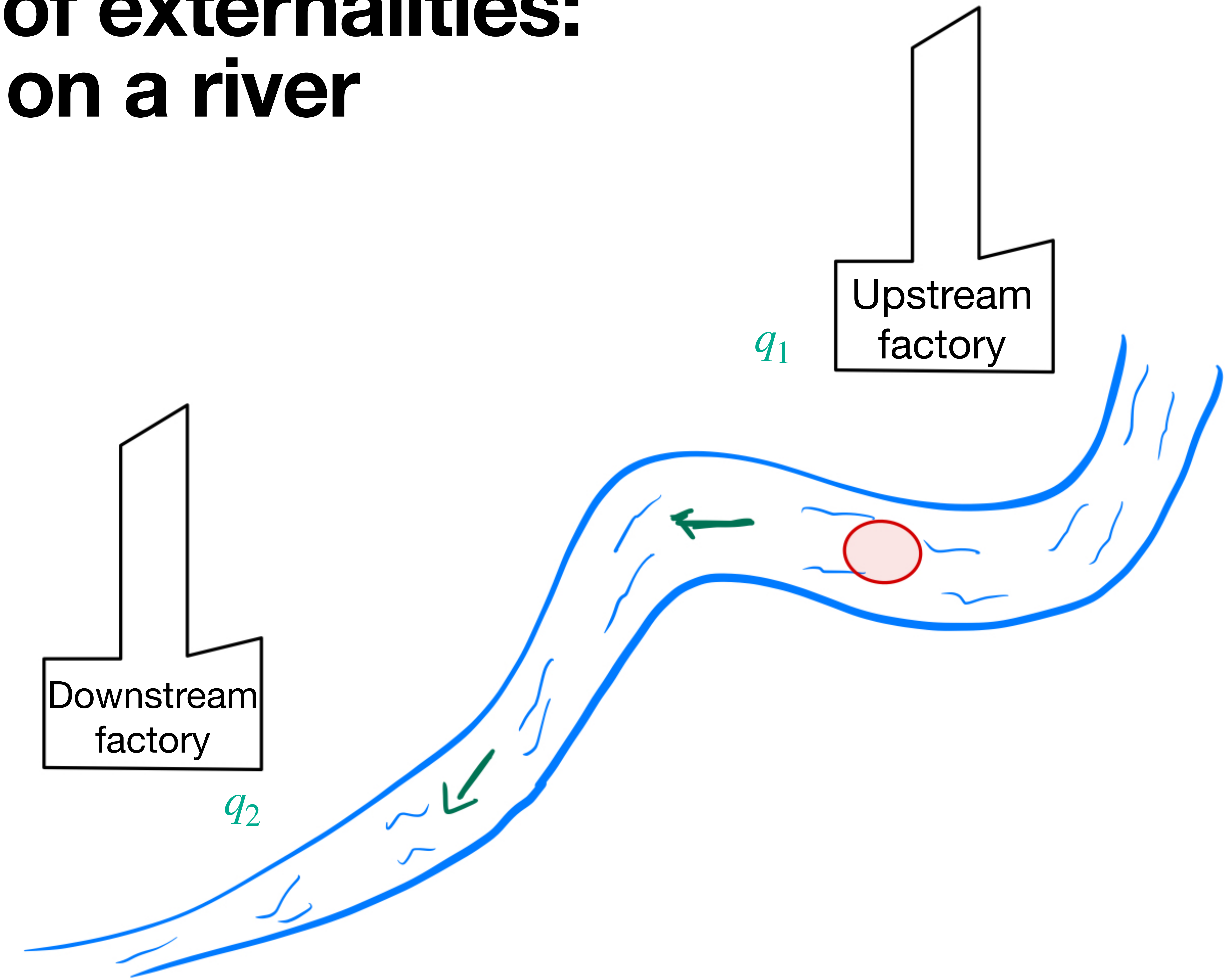
Mitigating Externalities while Learning

A. Scheid, A. Capitaine, E. Boursier, E. Moulines, M. Jordan, A. Durmus

NeurIPS 2024

Example of externalities: factories on a river



Example: factories on a river

Upstream utility: $\pi_1(q_1)$ increases with q_1

Downstream utility: $\pi_2(q_1, q_2)$ increases with q_2 and decreases with q_1

externality

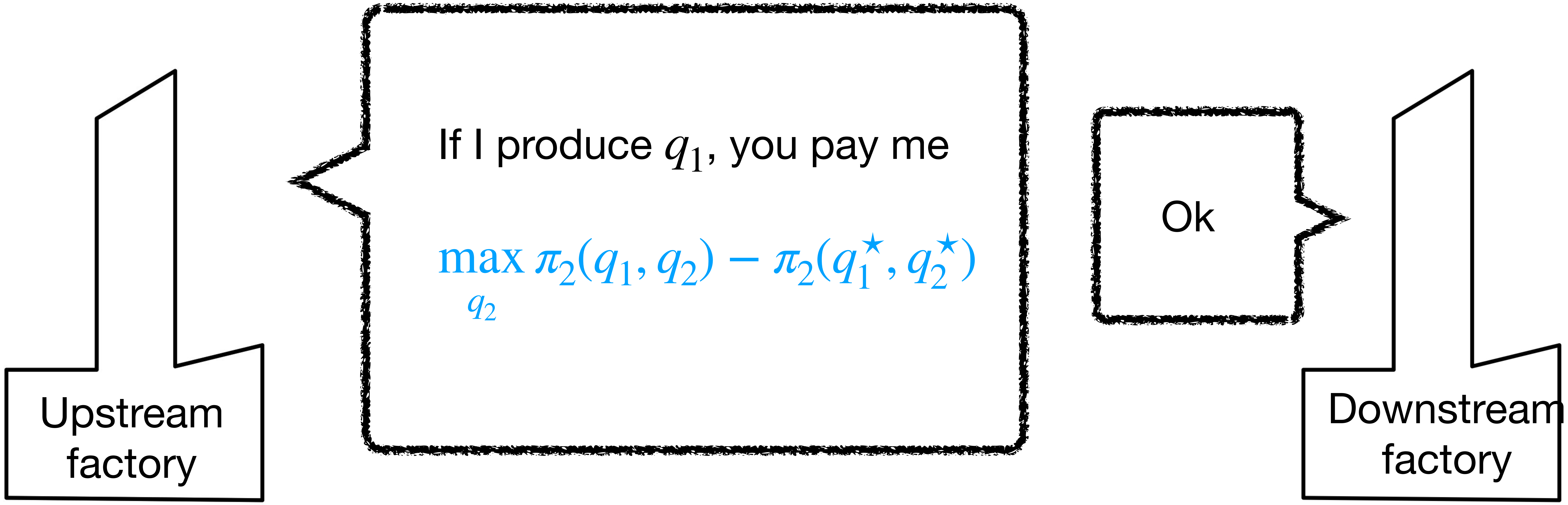
Upstream optimal strategy: $q_1^\star = \arg\max_q \pi_1(q)$

Downstream optimal strategy: $q_2^\star = \arg\max_q \pi_2(q_1^\star, q)$

Social inefficiency: $\pi_1(q_1^\star) + \pi_2(q_1^\star, q_2^\star) < max_{q_1,q_2}\{\pi(q_1) + \pi(q_1, q_2)\}$

Recovering social optimum

Idea: allowing proprietary rights to restore social efficiency

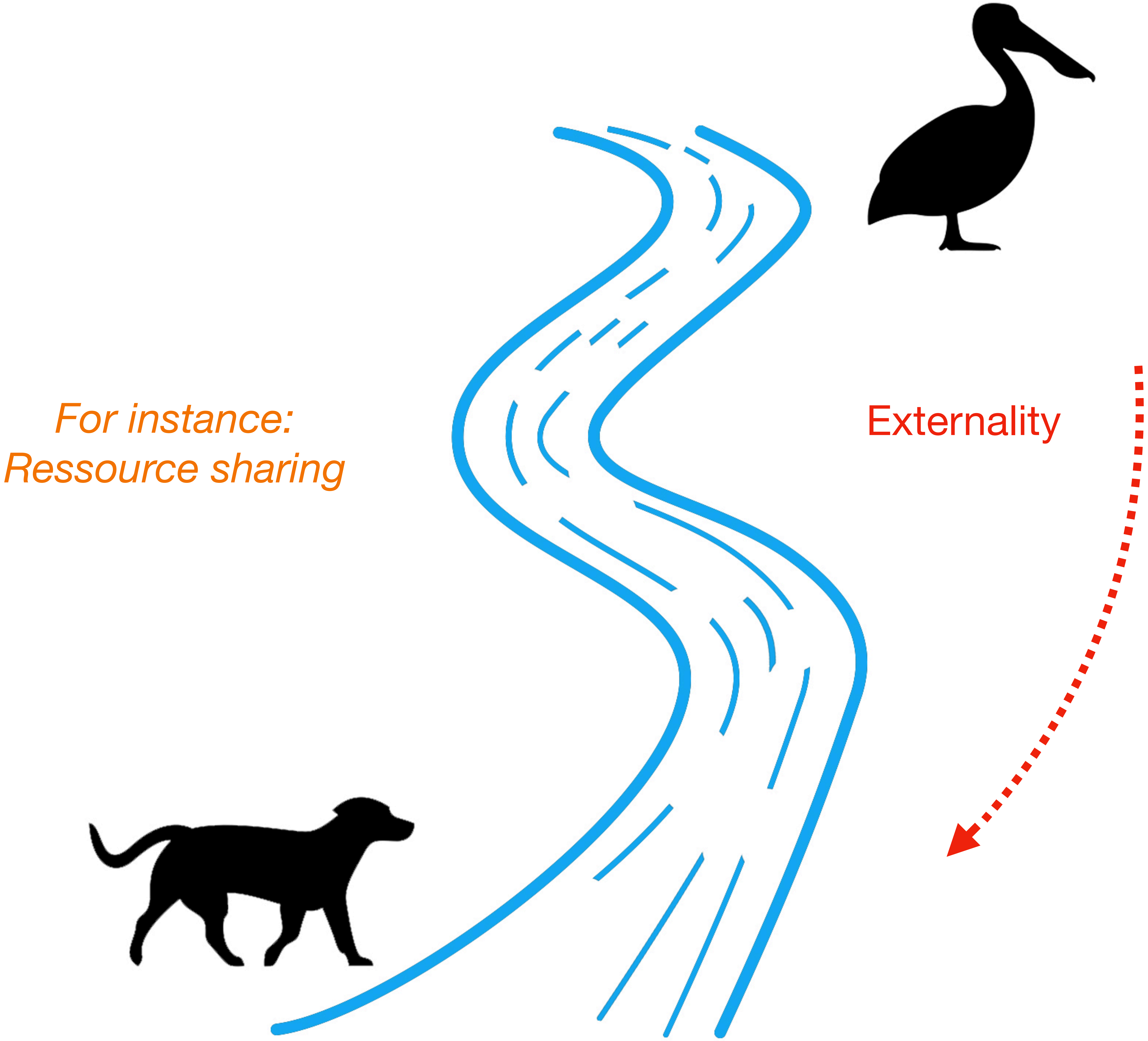


→ recover social optimum $\max_{q_1,q_2} \pi_1(q_1) + \pi_2(q_1, q_2)$

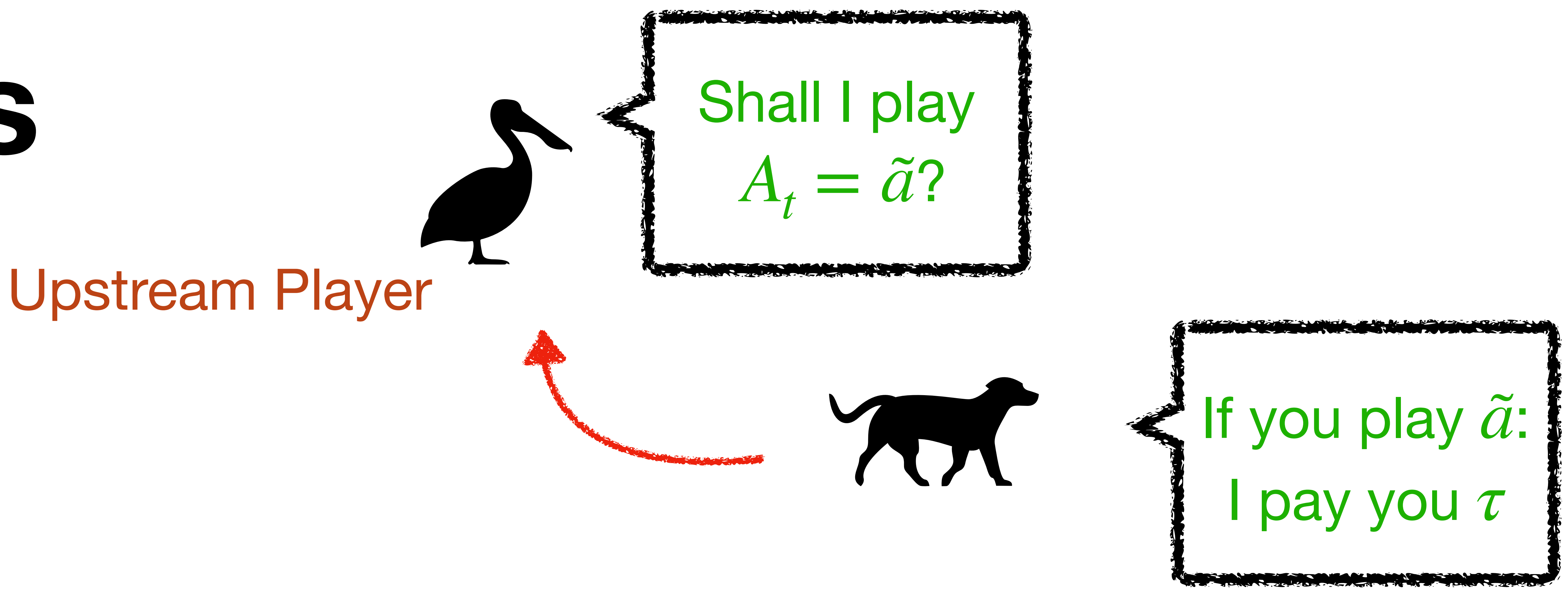
Issue: the players do not know their utilities!

What if we learn utility functions over time?

We can recover social optimum through transfers


For instance:
Ressource sharing
Externality

Model with transfers

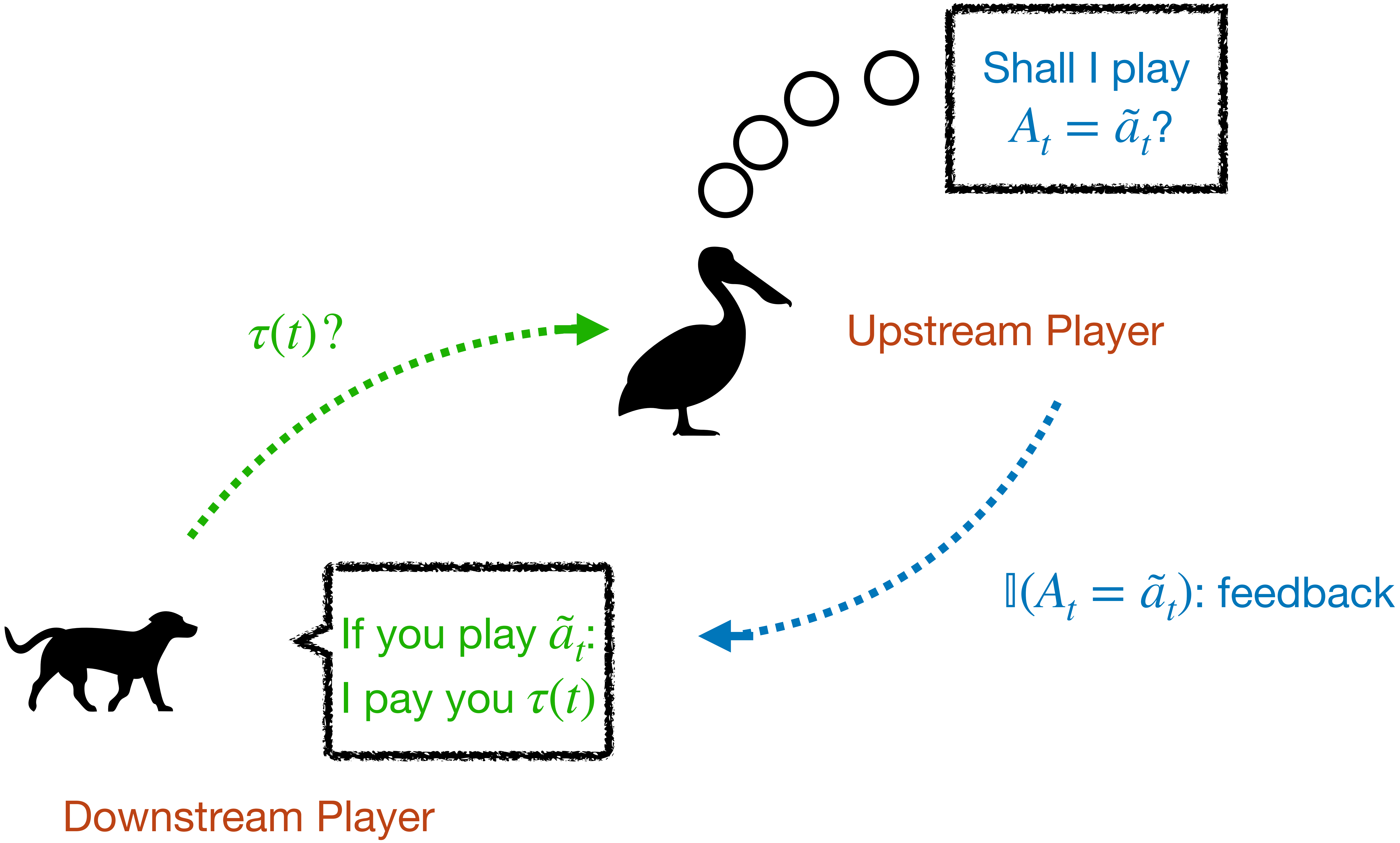


At each time t:

1. Downstream player proposes payment $(\tau, \tilde{a}) \in \mathbb{R}_+ \times [K]$

2. Upstream player: - observes $(\tau(t), \tilde{a}_t)$
   - plays $A_t \in [K]$
   - receives and observes: $Z_{A_t}(t) + \mathbf{1}_{A_t = \tilde{a}}\, \tau$

3. Downstream player: - plays $B_t \in [K]$
   - gets reward $X_{A_t,B_t}(t) - \mathbf{1}_{A_t = \tilde{a}}\, \tau$
   - observes separately $X_{A_t,B_t}(t)$ and $\mathbf{1}_{A_t = \tilde{a}}$


Shall I play
$A_t = \tilde{a}_t$?
$\tau(t)$?
Upstream Player
$\mathbb{I}(A_t = \tilde{a}_t)$: feedback
If you play $\tilde{a}_t$:
I pay you $\tau(t)$
Downstream Player

Designing no regret strategies

Upstream player is a no-regret learner

Downstream policy

Idea: run batched binary searches to find $\tau_a^\star$: *the minimal incentive* to have $A_t = a$

- Propose payment (a, τ) for T^α successive time steps
- Observe $T^{\neq}$ the number of times upstream did not pull a

Using the **no-regret assumption**, w.h.p.

- If $T^{\neq} > CT^{\alpha\kappa+\beta}$, then $\tau_a^\star > \tau - \dfrac{1}{T^\beta}$
- If $T^{\neq} < T^\alpha - CT^{\alpha\kappa+\beta}$, then $\tau_a^\star < \tau + \dfrac{1}{T^\beta}$

For any β s.t. $\alpha\kappa + \beta < \alpha$

Downstream policy

Theorem

For a good choice of parameters:

$$\mathscr{R}^{\mathrm{d}}(T) = o(T)$$

$\rightarrow$ most of the regret is due to *waiting for the upstream player learning*

$\rightarrow$ the faster does the upstream player learns, the better for the downstream one

$\rightarrow$ upstream and downstream players can typically use UCB as a subroutine

Conclusion

Summary:

- Study a repeated two player games with an upstream/downstream relation
- We propose a downstream algorithm that works for general upstream policies

Direct extensions:

- Instance dependent bounds
- Anytime policy
- Extension to linear contextual case

Perspectives

Higher level questions:

- More general interactions between multiple learning agents
- Propose *black box independent* strategies
- Potential long term strategic manipulations

Thank you!