HICOMM: HIERARCHICAL COMMUNICATION FOR COOPERATIVE MULTI-AGENT RL

Anonymous authors
Paper under double-blind review

ABSTRACT

Communication is essential for coordination in cooperative multi-agent reinforcement learning, yet existing protocols often overlook how the structure of the observation space can guide information exchange. We identify a shared hierarchical structure in many cooperative tasks: entities are organized into publicly indexed groups, while each agent observes only a subset of them. We formalize this setting as a *hierarchically indexed Markov decision process* (HI-MDP) and introduce HICOMM, a **Hi**erarchical **Comm**unication protocol that exploits the shared index for targeted information retrieval. Each receiver sends a request to a single teammate, chosen by the teammate's position in the hierarchy, and the teammate computes its reply from its local observation of one entry. Across cybersecurity, unit-control, and football benchmarks, HICOMM achieves comparable or better performance than recent learned communication baselines while substantially reducing communication cost. Our theoretical analysis further shows that, compared with the vanilla approach (flat communication), which does not use the hierarchical index, HICOMM has the same learning error bound, and its reward guarantee differs from that of flat communication only by the gap between the two protocols' optimal returns. However, the communication complexity of HICOMM is only $1/(N-1)$ of that of flat communication, where N is the number of agents. Such a theoretical analysis backs up our empirical findings.

1 INTRODUCTION

Cooperative multi-agent reinforcement learning (MARL) studies how a team of agents, each observing only part of the environment, can jointly maximize a shared return, with applications in cybersecurity, unit control, and team sports. Compared with single-agent reinforcement learning (RL), the main difficulty is that each agent must act on its own local observation, while the information needed for coordination may be spread across its teammates. For example, in the CAGE Challenge 4 (CC4) benchmark (Kiely et al., 2025), six attackers act in a simulated enterprise network whose hosts are grouped into nine subnets. An attacker learns the features of a host, such as whether it is reachable, only after discovering it, so its choice of next target depends on what its teammates have discovered in other subnets. Communication reduces this difficulty: agents that exchange what they observe need not infer the hidden state from their teammates' actions alone, and learned communication has improved coordination in many cooperative tasks (Sukhbaatar et al., 2016; Foerster et al., 2016; Jiang & Lu, 2018; Das et al., 2019; Hu et al., 2024).

Learned communication methods decide when to send a message (Singh et al., 2019; Jiang & Lu, 2018; Kim et al., 2019; Zhang et al., 2019; 2020), to whom (Das et al., 2019; Ding et al., 2020; Niu et al., 2021; Sun et al., 2024a), and what to send (Wang et al., 2020; Sun et al., 2024b; Li & Zhang, 2024), and many of them use these decisions to reduce communication cost. These methods change the protocol, but none of them uses the structure of the observation space to address messages. In many cooperative tasks, this space has a *group–entity hierarchy*, in which entities are organized into groups. The groups and entities are the attackers' zones of subnets and their hosts in the example above, unit types and units in unit control, and roles and players in team sports. All agents use the same group–entity index but observe different entities, so a receiver can use the index to choose which teammate to ask, and the teammate can use it to name the entry that it returns, a group and an entity in that group. We call communication in which each request reaches all teammates and each

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

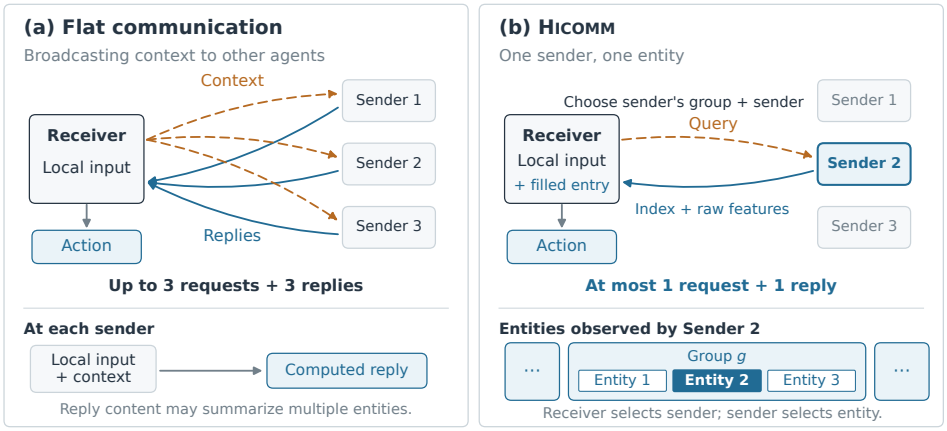


Figure 1: HiComm vs. flat communication, both two-stage protocols. A flat request reaches all three candidate senders and each may reply. A HiComm request reaches one sender, chosen leveraging hierarchy, and replies about one entity that it observes: in the lower-right inset, Sender 2 selects a group g and then Entity 2 of g.

of them may reply *flat*; it does not use the hierarchy, and in the example it can carry up to thirty requests and thirty replies per step. This raises our central question:

> *Can we leverage the hierarchical structure of the observation space to enable efficient communication without sacrificing performance?*

We answer this question affirmatively with HiComm, a hierarchical two-stage communication protocol. Our contributions are threefold:

- **Structure and protocol.** We identify a new structure shared by many cooperative tasks: the entities of the observation space are organized into groups, and the resulting group–entity index is common to all agents, although each agent observes only some of the entities. We formalize this structure as the *hierarchically indexed Markov decision process* (HI-MDP). Based on the HI-MDP, we propose HiComm, a simple two-stage communication protocol that leverages the hierarchy. Each receiver uses the index to ask one teammate, and this teammate uses the index to return one entry that it observes, which the receiver writes into its own observation. Both choices are learned selections, first over groups and then over teammates or entities, and are trained end to end with proximal policy optimization (PPO) (Schulman et al., 2017).
- **Evaluation.** We evaluate HiComm on diverse cooperative multi-agent benchmarks and compare it with state-of-the-art learned communication methods. HiComm outperforms these baselines in team reward in almost all settings while substantially reducing communication cost. These results show that leveraging the hierarchical structure of the observation space improves both performance and communication efficiency. Ablation studies further demonstrate the robustness of HiComm to errors in the group–entity index and validate its key design choices.
- **Theory.** We also deliver theoretical results on the convergence rate of learning under both protocols. We formulate learning under each protocol as a multi-agent constrained MDP with one communication constraint per agent, and prove that the average return of a primal–dual learner is at least the protocol's optimal return minus a vanishing error, while each agent's average communication cost exceeds its budget by at most a vanishing amount. These results provide theoretical support for our empirical findings.

2 Related Work

Learned communication conditions each agent's policy on messages from its teammates, as in CommNet, DIAL, and TarMAC (Sukhbaatar et al., 2016; Foerster et al., 2016; Das et al., 2019), which communicate at every step. Later work reduces communication: CACOM combines context broadcasts with selective, receiver-conditioned, individual replies (Li & Zhang, 2024); T2MAC

learns when and with whom to communicate (Sun et al., 2024a); ExpoComm spreads information over exponential graphs at linear or near-linear cost (Li et al., 2025). Emergent communication studies how agents come to share a protocol of discrete symbols from task reward alone (Lazaridou et al., 2017; Havrylov & Titov, 2017); Nikolaus (2024) adds a feedback channel from listener to speaker, which makes the protocol robust to channel noise. None of these uses the observation space's structure to address messages: a message is a learned code over the whole observation, even when attention over its entities computes it, and selection, where present, picks a link, not an entry. HICOMM uses the HI-MDP's shared group–entity index to target requests and to name the returned entry, which reduces the maximum number of messages delivered per step from quadratic in the number of agents under flat communication, such as CACOM, to linear.

The HI-MDP extends factored and object-oriented Markov decision processes (MDPs) (Boutilier et al., 2000; Diuk et al., 2008) with groups of entities under an index that all agents share (Remark 3.2). Entity-based reinforcement learning processes the entities of a state with relational attention (Zambaldi et al., 2019), entity-wise value factorization (Iqbal et al., 2021), or transformers over entity tokens (Hu et al., 2021); HICOMM instead uses the shared index as an address for communication. Other work structures the dependencies among agents rather than the information inside observations: DHCG learns directed acyclic communication graphs (Liu et al., 2023), and SeqComm orders the agents' decisions and passes the actions of earlier-deciding agents to later ones (Ding et al., 2024). In Where2comm, each agent broadcasts a request map of the regions where its confidence is low, and teammates send features only for requested regions (Hu et al., 2022); a HICOMM receiver instead asks one teammate, which chooses which of its observed entries to return. Hierarchical RL uses hierarchy for temporally extended actions (Sutton et al., 1999); HICOMM uses it to organize information access within a step.

Decentralized partially observable MDPs (Dec-POMDPs) with explicit communication formalize decision-making from distributed observations (Oliehoek & Amato, 2016; Goldman & Zilberstein, 2004; Pynadath & Tambe, 2002); the common-information approach (Nayyar et al., 2013) describes the agents' decision rules for each step through the information they share; the rules of all agents form a step policy (Section 6), and approximate information states relate prediction errors to decision quality (Subramanian et al., 2022; Kao & Subramanian, 2022). Our analysis uses constrained MDPs (Altman, 1999), in their multi-agent form (Lu et al., 2021), and primal–dual policy optimization (Stradi et al., 2025; Luo et al., 2021) to separate the return lost to the protocol from learning error.

3 PROBLEM SETUP

3.1 COOPERATIVE DECISION PROCESS AND THE HI-MDP

We consider a cooperative decentralized partially observable Markov decision process (Dec-POMDP) (Oliehoek & Amato, 2016) with deterministic observations: $N \geq 2$ agents $i, j = 1, \ldots, N$, horizon H with steps $h = 1, \ldots, H$, state space $\mathcal{S}$, action spaces $\mathcal{A}^i$ with joint space $\mathcal{A} = \prod_{i=1}^{N} \mathcal{A}^i$, and initial state distribution b_0 of s_1. At step h, agent i observes $o_h^i = O_h^i(s_h)$, where O_h^i is its observation function, and takes $a_h^i \in \mathcal{A}^i$; the team receives the shared reward $r(s_h, \mathbf{a}_h) \in [0, 1]$ for the joint action $\mathbf{a}_h = (a_h^1, \ldots, a_h^N)$, and the next state is drawn from the transition kernel P as $s_{h+1} \sim P(\cdot \mid s_h, \mathbf{a}_h)$.

Definition 3.1 (HI-MDP)**.** A *hierarchically indexed Markov decision process* (HI-MDP) is a cooperative Dec-POMDP with the following structure.

1. *Index.* There is a finite set of groups $\mathcal{G}$, and each group $g \in \mathcal{G}$ contains a finite set of entities $\mathcal{E}_g$. An *entry* is a pair (g, e) with $e \in \mathcal{E}_g$, and it names the same entity for every agent.

2. *State and observations.* At step h, the state s_h determines, for each agent i, a mask M_h^i, which is the set of entries that agent i observes, and features $o_h^i[g, e]$ for each entry $(g, e) \in M_h^i$. For $(g, e) \notin M_h^i$, we set $o_h^i[g, e] = 0$ and call (g, e) an *empty entry* for agent i. Agent i observes M_h^i and o_h^i.

Table 1: The HI-MDP formulation of each benchmark. Agent i is the observing agent.

	CC4 (native)	**SMACv2** (semantic)	**GRF** (semantic)
Groups $g \in \mathcal{G}$	6 attacker zones; internet	ally unit types; opponents	role groups; opponents
Entities $e \in \mathcal{E}_g$	host slots of the group	units of the group	players of the group
Features $o_h^i[g, e]$	access and recon levels, discovered neighbors, reachability, known deception (5)	visibility, relative distance and position, health, shield (`protoss`), unit type (8–9)	position, direction, speed, distance to i, tiredness (7)
Mask M_h^i	hosts that i has discovered	alive units in i's sight range	players in i's sensing radius
Agents $\widehat{e} \in \widehat{\mathcal{E}}_g$	the zone's attacker; none for the internet	the agents of the unit type; none for opponents	the agents of the role group; none for opponents

3. *Agents.* Each group g also contains a set of agents $\widehat{\mathcal{E}}_g \subseteq \{1, \ldots, N\}$, which may be empty, and every agent belongs to exactly one group. Receivers use pairs $(\widehat{g}, \widehat{e})$ with $\widehat{e} \in \widehat{\mathcal{E}}_{\widehat{g}}$ to locate senders, and senders use entries (g, e) to name the information they return.

Only an agent whose mask contains an entry can supply its features. The entities of group g that agent i observes at step h form $\mathcal{E}_{g,h}^i = \{e \in \mathcal{E}_g : (g, e) \in M_h^i\}$, and the groups in which it observes an entity form $\mathcal{G}_h^i = \{g \in \mathcal{G} : \mathcal{E}_{g,h}^i \neq \varnothing\}$. The groups that contain a teammate of i form $\widehat{\mathcal{G}}^i = \{g \in \mathcal{G} : \widehat{\mathcal{E}}_g \setminus \{i\} \neq \varnothing\}$.

Remark 3.2. Factored, object-oriented, and entity-based models describe the state by variables, objects, or typed entities (Boutilier et al., 2000; Diuk et al., 2008; Iqbal et al., 2021). The HI-MDP adds groups above the entities and makes the resulting index common to all agents: entry (g, e) names the same entity for every agent, including an agent whose mask does not contain it.

Remark 3.3. An environment is an HI-MDP when it provides groups for its entities, in one of two ways. In a *native* HI-MDP, the environment is itself hierarchical: CC4 (Kiely et al., 2025) divides its hosts into nine subnets of 16 host slots; each attacker acts in its own zone, a single subnet or, for one attacker, the three headquarters subnets, and the internet subnet belongs to no zone. A group is the zone of one attacker or the internet, and an entity is a host slot in that group. Each zone contains its attacker as its only agent, and the internet contains no agent. In a *semantic* HI-MDP, we group the entities by a label that the environment exposes: by unit type in SMACv2 (Ellis et al., 2023) and by on-pitch role in Google Research Football (GRF) (Kurach et al., 2020), with the opponents as one further group. In both, an allied group contains the agents whose units or players belong to it, and the opponent group contains no agent. Features can differ between observers: some SMACv2 and GRF features, such as distance, are observer-relative, and CC4 features are each attacker's accumulated knowledge of the host. Table 1 lists the groups, entities, features, masks, and agents of each benchmark.

3.2 Two-Stage Communication Protocol

Following Ding et al. (2020) and Li & Zhang (2024), we consider *two-stage communication protocols*: at every step, the agents communicate in two rounds, a request round and a reply round, with error-free delivery. The agent that asks is the *receiver*, the agent that answers is the *sender*, and an agent may be both; a request together with its replies is an *exchange*. Agent i's local history I_h^i holds its observations and masks through step h, its past actions, and the messages it sent or received at earlier steps. Two-stage communication protocols differ in who receives a request and what a reply may contain. *Flat* communication, which does not use the index, is the vanilla two-stage protocol:

1. *Request.* Agent i broadcasts a context q_h^i, computed from I_h^i, to all teammates, or stays silent, $q_h^i = \perp$; throughout, $\perp$ means that nothing is sent or chosen.
2. *Reply.* Each agent j that received q_h^i decides from (I_h^j, q_h^i) whether to reply. A reply $m_{h,j\to i}$ can be any message; a slot without a reply holds $\perp$.

Receiver i collects the replies $m_h^i = (m_{h,j\to i})_{j\neq i}$ and samples its action from the distribution π_h^i

$$a_h^i \sim \pi_h^i(\cdot \mid I_h^i, m_h^i). \tag{3.1}$$

We write $\mathcal{F}$ for flat communication, $\mathcal{H}$ for HICOMM (Section 4), and $\mathrm{alg} \in \{\mathcal{H}, \mathcal{F}\}$ for either. A *policy* π of protocol alg specifies, for every agent and step, a *request rule* from I_h^i, a *reply rule* from the agent's local history and each request it receives, and an *action rule* π_h^i from (I_h^i, m_h^i) as in Eq. (3.1). Agents may also use shared random draws that are independent of the environment. The set of such policies is the *policy class* Π_{alg} of protocol alg, and a policy $\pi \in \Pi_{\mathrm{alg}}$ has expected return $V^\pi = \mathbb{E}_{b_0}^\pi\big[\sum_{h=1}^H r(s_h, \mathbf{a}_h)\big] \in [0, H]$.

3.3 COMMUNICATION COST AND BUDGET

A transmitted request costs b_q bits and a transmitted reply b_r bits, with the same positive integer widths b_q, b_r under both protocols. The null reply of HICOMM (Section 4.1), which a sender that observes no entry returns and which carries no features, still costs b_r bits; silence and an empty reply slot cost nothing. The step cost c_h is the number of bits delivered at step h, where a message counts once for each agent that receives it; the step cost c_h^i of agent i counts the bits delivered in the exchange that receiver i starts, so $c_h = \sum_{i=1}^N c_h^i$. The *communication cost* of a policy is $C^\pi = \mathbb{E}_{b_0}^\pi[\sum_{h=1}^H c_h]$, and that of agent i is $C^{\pi,i} = \mathbb{E}_{b_0}^\pi[\sum_{h=1}^H c_h^i]$. The *communication capacity* α_{alg} of protocol alg is the largest team step cost c_h it allows, and its *communication budget* $B_{\mathrm{alg}} = H\alpha_{\mathrm{alg}}$ is the largest number of bits an episode can use, so $C^\pi \le B_{\mathrm{alg}}$ for every $\pi \in \Pi_{\mathrm{alg}}$. The optimal return of protocol alg is

$$V_{\mathrm{alg}}^\star = \sup\big\{V^\pi : \pi \in \Pi_{\mathrm{alg}}\big\}. \tag{3.2}$$

4 METHODOLOGY: HIERARCHICAL COMMUNICATION (HICOMM)

HICOMM restricts the two-stage flat protocol of Section 3.2. In HICOMM, each receiver asks one teammate and receives one entry. Both choices use the index (Figure 1), and both choices are learned (Algorithm 1).

4.1 PROTOCOL

1. *Request.* Receiver i either stays silent or sends a query q_h^i to one teammate $j_h^i \neq i$. The receiver computes q_h^i from I_h^i. We call j_h^i the sender.
2. *Reply.* Sender $j = j_h^i$ uses (I_h^j, q_h^i) to choose an entry $(g_h^i, e_h^i) \in M_h^j$. It replies with the index and the features of this entry, $\big((g_h^i, e_h^i),\, o_h^j[g_h^i, e_h^i]\big)$. If $M_h^j = \varnothing$, the sender sends the *null reply*, which carries no features.

In one step, HICOMM delivers at most N requests and N replies. The flat protocol delivers up to $N(N-1)$ requests and $N(N-1)$ replies. Hence $\alpha_{\mathcal{F}} = N(N-1)(b_q+b_r)$ and $\alpha_{\mathcal{H}} = N(b_q+b_r) = \alpha_{\mathcal{F}}/(N-1)$. We call the reply above a *raw-feature reply*. When an entity is described by high-dimensional features such as an image, an *encoded reply* can replace the raw-feature reply. The encoded reply compresses the sender's observation of the entry into b_r bits.

4.2 IMPLEMENTATION

Query and keys. The receiver computes the encoding $u_h^i = f_{\mathrm{enc}}(o_h^i, M_h^i)$ and the query $q_h^i = \tanh(f_{\mathrm{q}}(u_h^i))$. Each choice compares the query with one key per candidate. The receiver locates the sender by a pair $(\widehat{g}, \widehat{e})$ with $\widehat{e} \in \widehat{\mathcal{E}}_{\widehat{g}}$, and the sender names the returned entry by (g, e) with $e \in \mathcal{E}_g$ (Definition 3.1), so each side has a group key and a member key. The group keys $\widehat{\kappa}(\widehat{g})$ and $\kappa(g)$ are learned vectors, and $\widehat{\kappa}(\perp)$ is the key of silence. The receiver's agent key $\widehat{f}_{\mathrm{e}}(\widehat{g}, \widehat{e}; u_h^i)$ is a multilayer perceptron (MLP) applied to the receiver's encoding and learned embeddings of $\widehat{g}$ and $\widehat{e}$; it does not read the teammate's features, which the receiver does not observe. The sender's entity key $f_{\mathrm{e}}(o_h^j[g, e])$ is an MLP applied to the features that sender j observes for entry (g, e). Each MLP shares its weights across candidates. Every key has the same dimension as q_h^i. We call

Algorithm 1 HiComm training. At each step h, lines 2–6 run for every receiver i in parallel.

Input: Modules $\mathbf{f}$, selection function sel (Section 4.2), policies π_{pol}, critic, buffer $\mathcal{D} = \varnothing$, updates T.

1: **repeat**
2: $u_h^i \leftarrow f_{\text{enc}}(o_h^i, M_h^i)$; $q_h^i \leftarrow \tanh(f_{\text{q}}(u_h^i))$. — query
3: $\widehat{g}_h^i \leftarrow \text{sel}(q_h^i, \widehat{\kappa}, \widehat{\mathcal{G}}^i \cup \{\perp\})$; if $\widehat{g}_h^i = \perp$, go to line 6. — select
4: $\widehat{e}_h^i \leftarrow \text{sel}(q_h^i, \widehat{f}_{\text{e}}(\widehat{g}_h^i, \cdot\,; u_h^i), \widehat{\mathcal{E}}_{\widehat{g}_h^i} \setminus \{i\})$; $j_h^i \leftarrow \widehat{e}_h^i$. — sender
5: Sender $j = j_h^i$: $g_h^i \leftarrow \text{sel}(q_h^i, \kappa, \mathcal{G}_h^j)$, $e_h^i \leftarrow \text{sel}(q_h^i, f_{\text{e}}(o_h^j[g_h^i, \cdot]), \mathcal{E}^j_{g_h^i, h})$, or null. — reply
6: $a_h^i \sim \pi_{\text{pol}}^i(\cdot \mid \widetilde{o}_h^i)$, with $\widetilde{o}_h^i$ from Eq. (4.1). — write and act
7: Execute $\mathbf{a}_h$; store the transition, the selections, and their Gumbel draws in $\mathcal{D}$.
8: If $\mathcal{D}$ holds a batch, update $\mathbf{f}$, π_{pol}, and the critic by PPO with the auxiliary loss; $\mathcal{D} \leftarrow \varnothing$.
9: **until** T updates are done

$\mathbf{f} = (f_{\text{enc}}, f_{\text{q}}, \widehat{\kappa}, \widehat{f}_{\text{e}}, \kappa, f_{\text{e}})$ the *communication modules*. All agents share $\mathbf{f}$. On SMACv2 and GRF, all agents also share the policy π_{pol}^i.

Selection. Each choice calls the selection function $\text{sel}(q, \psi, \mathcal{C})$. The function takes a query q, a finite set $\mathcal{C}$ of candidates, and a map ψ that gives the key $\psi(c)$ of each candidate c. It scores each candidate $c \in \mathcal{C}$ by $\langle q, \psi(c) \rangle$ and samples one candidate from the softmax of the scores. If $\mathcal{C}$ is empty, it returns $\perp$. We implement sel with straight-through Gumbel-Softmax (Jang et al., 2017). The forward pass returns the candidate with the largest Gumbel-perturbed score as a one-hot vector. The backward pass uses the Jacobian of the softmax of the perturbed scores.

Choosing the sender and the entry. Both sides call sel twice (Algorithm 1, lines 3–5). The receiver chooses silence or a group $\widehat{g}_h^i \in \widehat{\mathcal{G}}^i$ with the keys $\widehat{\kappa}$, and then the sender $j_h^i = \widehat{e}_h^i \in \widehat{\mathcal{E}}_{\widehat{g}_h^i} \setminus \{i\}$ with the keys $\widehat{f}_{\text{e}}(\widehat{g}_h^i, \cdot\,; u_h^i)$. If this set has one element, as in CC4, where each zone contains only its attacker, the second choice has one candidate. The sender chooses a group $g_h^i \in \mathcal{G}_h^j$ with the keys κ, and then an entity $e_h^i \in \mathcal{E}^j_{g_h^i, h}$ with the keys $f_{\text{e}}(o_h^j[g_h^i, \cdot])$. If $M_h^j = \varnothing$, then $\mathcal{G}_h^j = \varnothing$, sel returns $\perp$, and the sender sends the null reply.

Written observation. Entries are common to all agents. Receiver i therefore writes the reply into its own observation. It acts on the *written observation*

$$\widetilde{o}_h^i[g, e] = \begin{cases} o_h^{j_h^i}[g, e], & (g, e) = (g_h^i, e_h^i) \notin M_h^i, \\ o_h^i[g, e], & \text{otherwise.} \end{cases} \tag{4.1}$$

After silence or a null reply, $\widetilde{o}_h^i = o_h^i$.

Training. We train $\mathbf{f}$, π_{pol}, and a critic with PPO (Schulman et al., 2017). We call the gradient of the actor's PPO loss with respect to $\mathbf{f}$ the *task gradient*. The loss does not contain the features of the teammates and groups that are not chosen. The task gradient therefore gives no signal about these candidates. We add an auxiliary loss to supply this signal. The auxiliary loss is computed during training from the observations of all agents. Its cross-entropy targets weight each candidate by the positive part of the features that the candidate can return for entries the receiver does not observe. Appendix B.1 gives further details.

5 Experiments

We evaluate HiComm on the three benchmarks of Table 1 and ask two questions. **(RQ1) Performance:** how much of the reward gap between PartialObs, which does not communicate, and FullObs, which receives the concatenated observations of all teammates, does HiComm close, and does it match or exceed the team reward (the mean episode return) of learned communication baselines? **(RQ2) Communication cost:** how much does HiComm communicate relative to these baselines and to FullObs?

Table 2: Team reward and communication cost (KiB per receiver per episode), mean ± standard deviation over three training seeds. Bold marks the two highest mean rewards and the two lowest mean costs per backbone and scenario; PARTIALOBS, which transmits nothing, always has the lowest cost. Within a scenario, the cost of FULLOBS varies only with episode length.

		CC4		SMACv2 5v5						GRF 4v3	
				protoss		terran		zerg			
Algorithm	Method	Reward	Cost	Reward	Cost	Reward	Cost	Reward	Cost	Reward	Cost
IPPO	FULLOBS	$\mathbf{1{,}724 \pm 185}$	$7{,}017.2 \pm 0$	$\mathbf{16.27 \pm 1.68}$	242.1 ± 6.5	$\mathbf{16.59 \pm 1.38}$	199.6 ± 15.3	$\mathbf{14.82 \pm 1.06}$	141.5 ± 12.4	$\mathbf{1.42 \pm 0.22}$	98.8 ± 7.3
IPPO	PARTIALOBS	$1{,}395 \pm 134$	$\mathbf{0 \pm 0}$	14.01 ± 1.59	$\mathbf{0 \pm 0}$	14.23 ± 1.30	$\mathbf{0 \pm 0}$	10.80 ± 0.52	$\mathbf{0 \pm 0}$	1.05 ± 0.03	$\mathbf{0 \pm 0}$
IPPO	CACOM	$1{,}515 \pm 73$	33.5 ± 1.2	15.60 ± 0.60	20.8 ± 1.1	14.40 ± 0.76	17.4 ± 1.1	12.18 ± 0.21	6.4 ± 0.9	1.15 ± 0.08	17.2 ± 1.1
IPPO	T2MAC	$\mathbf{1{,}620 \pm 248}$	79.8 ± 12.0	14.95 ± 1.35	41.3 ± 2.3	14.40 ± 0.22	80.8 ± 4.6	12.07 ± 1.25	63.6 ± 2.8	1.15 ± 0.12	20.3 ± 0.4
IPPO	EC	$1{,}454 \pm 299$	27.4 ± 4.2	14.42 ± 2.20	32.4 ± 3.9	14.76 ± 1.97	30.0 ± 2.4	12.21 ± 1.54	19.7 ± 2.2	1.18 ± 0.13	12.0 ± 1.6
IPPO	HICOMM (ours)	$1{,}572 \pm 107$	$\mathbf{10.4 \pm 1.0}$	$\mathbf{15.96 \pm 1.51}$	$\mathbf{6.1 \pm 0.8}$	$\mathbf{16.17 \pm 1.13}$	$\mathbf{4.9 \pm 1.1}$	$\mathbf{13.48 \pm 0.97}$	$\mathbf{3.7 \pm 0.3}$	$\mathbf{1.35 \pm 0.12}$	$\mathbf{6.4 \pm 0.6}$
MAPPO	FULLOBS	$\mathbf{2{,}163 \pm 129}$	$7{,}017.2 \pm 0$	$\mathbf{17.38 \pm 0.69}$	236.9 ± 8.6	$\mathbf{16.41 \pm 0.63}$	182.7 ± 14.6	$\mathbf{15.57 \pm 2.04}$	138.0 ± 12.0	$\mathbf{1.64 \pm 0.07}$	96.5 ± 4.7
MAPPO	PARTIALOBS	$1{,}048 \pm 150$	$\mathbf{0 \pm 0}$	14.77 ± 0.66	$\mathbf{0 \pm 0}$	12.98 ± 0.51	$\mathbf{0 \pm 0}$	12.11 ± 0.77	$\mathbf{0 \pm 0}$	1.15 ± 0.20	$\mathbf{0 \pm 0}$
MAPPO	CACOM	$1{,}669 \pm 162$	35.6 ± 1.9	15.34 ± 1.55	15.5 ± 1.0	14.48 ± 0.97	8.6 ± 0.3	13.84 ± 2.39	5.9 ± 1.3	1.35 ± 0.22	17.7 ± 0.4
MAPPO	T2MAC	$1{,}694 \pm 180$	83.0 ± 17.6	15.55 ± 0.64	36.6 ± 1.2	13.78 ± 0.45	161.0 ± 9.0	12.50 ± 0.81	24.8 ± 1.5	1.26 ± 0.17	22.0 ± 0.6
MAPPO	EC	$1{,}568 \pm 157$	37.1 ± 4.1	15.85 ± 0.56	35.8 ± 4.5	14.05 ± 0.55	39.0 ± 10.2	12.17 ± 0.53	21.9 ± 1.2	1.31 ± 0.08	8.5 ± 0.7
MAPPO	HICOMM (ours)	$\mathbf{1{,}917 \pm 158}$	$\mathbf{10.5 \pm 0.1}$	$\mathbf{16.29 \pm 0.23}$	$\mathbf{5.1 \pm 0.2}$	$\mathbf{14.84 \pm 1.80}$	$\mathbf{5.5 \pm 0.4}$	$\mathbf{13.90 \pm 1.38}$	$\mathbf{3.5 \pm 0.1}$	$\mathbf{1.53 \pm 0.05}$	$\mathbf{6.6 \pm 0.1}$

5.1 EXPERIMENTAL SETUP

Benchmarks. In CC4, six attackers act against a fixed pretrained defender. From SMACv2 we use three 5v5 scenarios (`protoss`, `terran`, and `zerg`), and from GRF the scenario `academy_counterattack_hard` (4v3; three controlled agents). On SMACv2 and GRF, features such as distance are observer-relative, and a reply keeps them in the sender's frame. Appendix B gives implementation details, benchmark configurations, full results, and ablation setups.

Baselines, metrics, and training. We compare with three state-of-the-art learned communication baselines: CACOM (Li & Zhang, 2024), a learned instance of flat communication; T2MAC (Sun et al., 2024a), which selects communication links and transmits learned evidence; and EC, our implementation of the feedback-based emergent-communication protocol of Nikolaus (2024). All methods are trained end to end by PPO under two backbones: independent PPO (IPPO) (Schroeder de Witt et al., 2020), with local critics, and multi-agent PPO (MAPPO) (Yu et al., 2022), with centralized critics. In all reported runs, each HICOMM receiver may stay silent at any step. We report two metrics: team reward, the mean evaluation episode return, and communication cost, the size of the messages exchanged in one evaluation episode, in KiB per receiver.

Performance (RQ1). In all ten configurations of Table 2, HICOMM's mean reward lies between those of PARTIALOBS and FULLOBS and closes 52%–86% of the gap, $(R - R_{\mathrm{P}})/(R_{\mathrm{F}} - R_{\mathrm{P}})$ for the mean rewards $R, R_{\mathrm{P}}, R_{\mathrm{F}}$ of HICOMM, PARTIALOBS, and FULLOBS. It exceeds the mean reward of every communication baseline in nine of ten configurations; the exception is CC4 under IPPO, where T2MAC is higher. In four configurations (CC4 and `protoss` under IPPO; `terran` and `zerg` under MAPPO), the difference from the best baseline is smaller than both methods' standard deviations. The largest margins over the best baseline, 13%–14% of its mean reward, occur on GRF under both backbones and on CC4 under MAPPO.

Communication cost (RQ2). In all ten configurations, HICOMM's communication cost is lower than that of every communication baseline, by factors of 1.6–3.6 relative to CACOM, 3.2–29 relative to T2MAC, and 1.3–7.1 relative to EC. It is 0.15% of the FULLOBS cost on CC4, 2.2%–3.0% on SMACv2, and 6%–7% on GRF. Relative to CACOM, part of the saving follows from the protocol: a HICOMM receiver receives at most one reply per step, whereas under flat communication it can receive one from every teammate.

5.2 ABLATIONS

Design variants. In Table 3, replacing the raw-feature reply by a learned encoding lowers mean reward under both backbones, whether the encoding covers the sender's observation of all hosts (*Encode host matrix*) or only the selected host's five features, as the encoded reply of Section 4.1 does (*Encode selected entity*). Replacing the sender's two-step choice of the returned entry, a group and then a host in it, by one pick over all 144 (group, host) pairs (*Joint group/entity selection*) also lowers mean reward under both backbones, although this variant still returns one host.

Table 3: CC4 design and detach variants: team reward, mean over three training seeds. The reference is HICOMM's entry in Table 2.

Variant	IPPO	MAPPO
HICOMM (reference)	1,572	1,917
Encode host matrix	1,431	1,709
Encode selected entity	1,401	1,818
Joint group/entity selection	1,536	1,604
Detached from the task gradient		
All communication	1,517	1,648
Query	1,505	1,586

Table 4: CC4 reward under IPPO after a fraction of the hosts is moved to other host slots, without retraining.

Reassigned	0%	5%	10%	20%
Reward	1,572	1,549	1,524	1,501

Table 5: CC4 reward and communication cost under IPPO with k senders per request; mean over three training seeds.

Senders k	1	2	3	4
Reward	1,572	1,549	1,553	1,581
Cost (KiB)	10.4	19.7	28.3	38.8

Task gradient. The detach variants of Table 3 block the task gradient from all communication modules (*All communication*) or the query (*Query*). Detaching the query also blocks the gradient of the auxiliary loss, so in both variants the query receives no training signal. The reference, in which no module is detached, exceeds these variants by 55–67 under IPPO and 269–331 under MAPPO.

Reassigned hosts. HICOMM reads the groups from the environment; on CC4 the simulator determines the group of every host. This test measures the reward lost when part of this assignment is wrong at evaluation. We evaluate the three IPPO policies of Table 2 without retraining after 5%, 10%, or 20% of the hosts are moved to other host slots. A moved host keeps its features and its mask entry; only the entry (g, e) under which it appears changes (Appendix B.7). Mean reward falls from 1,572 to 1,549, 1,524, and 1,501, by 1.5%, 3.1%, and 4.5% (Table 4). All three decreases are smaller than the across-seed standard deviation of the reference (107). At 20%, HICOMM keeps 60% of its gain $R - R_{\mathrm{P}}$ over PARTIALOBS, closes 32% instead of 54% of the gap to FULLOBS, and stays above EC (1,454) but below CACOM (1,515). The sender's entity keys f_{e} read only a host's features, so they score a moved host as before; the group keys κ and the actor were trained on the original group composition.

Several senders. This test asks whether restricting each receiver to one sender costs reward on CC4. Each receiver that does not stay silent asks $k \in \{2, 3, 4\}$ distinct teammates, each returns one entry by the unchanged reply rule, and the receiver writes every returned entry that it does not observe (Appendix B.7). A receiver then receives up to k replies per step, so each agent's step cost can reach $k(b_q + b_r)$, the capacity becomes $k\,\alpha_{\mathcal{H}}$, and $k = N - 1 = 5$ would reach the flat capacity $\alpha_{\mathcal{F}}$. Under IPPO, mean reward is 1,549, 1,553, and 1,581 for $k = 2, 3, 4$, against 1,572 for $k = 1$ (Table 5). The change is at most 1.5%, is not monotone in k, and its range (32) is less than a third of the reference's standard deviation. Cost grows almost linearly in k, by 8.6–10.5 KiB per added sender; at $k = 4$ (38.8 KiB) it exceeds that of CACOM (33.5). Extra replies thus raise the cost without a measurable change in reward, which supports the single-sender design.

6 THEORETICAL ANALYSIS

For the theoretical analysis, we introduce the step-policy MDP $\mathcal{M}$ of each protocol $\mathrm{alg} \in \{\mathcal{H}, \mathcal{F}\}$. It is a constrained MDP, built on an HI-MDP with finite state and action sets, whose state every agent knows (Nayyar et al., 2013). Details are deferred to Appendix A.

The step-policy MDP and learning. $\mathcal{M} = \big(\overline{\mathcal{S}}, (\overline{\mathcal{A}}_h)_{h=1}^{H}, \bar{r}, (\bar{c}^i)_{i=1}^{N}\big)$ has horizon H. At step h, instead of choosing an action directly, $\mathcal{M}$ chooses the rules that all agents follow at this step. These rules form a *step policy* $\Gamma_h = (\Gamma_h^1, \ldots, \Gamma_h^N)$, where Γ_h^i specifies, as deterministic functions of the part of agent i's local history that it did not choose and of the messages it receives, whether and what agent i requests, how it replies to each teammate, and which action it takes. The action set $\overline{\mathcal{A}}_h$ is the finite set of step policies of protocol alg at step h. The state is the sequence of step policies chosen so far, $\bar{s}_h = (\Gamma_1, \ldots, \Gamma_{h-1}) \in \overline{\mathcal{S}}_h = \overline{\mathcal{A}}_1 \times \cdots \times \overline{\mathcal{A}}_{h-1}$, with $\bar{s}_1 = \varnothing$ and $\overline{\mathcal{S}} = \bigcup_h \overline{\mathcal{S}}_h$, and

the transition $\bar{s}_{h+1} = (\bar{s}_h, \Gamma_h)$ is deterministic. The reward $\bar{r}(\bar{s}_h, \Gamma_h)$ and the cost $\bar{c}^i(\bar{s}_h, \Gamma_h)$, both in $[0,1]$, are the expected reward $r(s_h, \mathbf{a}_h)$ and the expected step cost c_h^i of agent i divided by its largest value α_{alg}/N at step h when the agents follow $\Gamma_1, \ldots, \Gamma_h$ in the HI-MDP; the learner does not know them. A policy π maps each state $\bar{s}_h$ to a distribution $\pi(\cdot \mid \bar{s}_h)$ on $\overline{\mathcal{A}}_h$: the agents draw $\Gamma_h \sim \pi(\cdot \mid \bar{s}_h)$ with shared randomness, so every agent knows $\bar{s}_h$, and each agent i applies Γ_h^i to its own local history and received messages. Let $\bar{A} \geq \max_h |\overline{\mathcal{A}}_h|$ and $\bar{S} \geq |\overline{\mathcal{S}}|$ bound the numbers of actions and states for both protocols.

The return and the cost of agent i of a policy π of $\mathcal{M}$, and the optimal return of protocol alg, are

$$\begin{gathered} \bar{V}^{\pi} = \mathbb{E}^{\pi}\Big[\sum_{h=1}^{H} \bar{r}(\bar{s}_h, \Gamma_h)\Big], \qquad \bar{C}^{\pi,i} = \frac{\alpha_{\mathrm{alg}}}{N}\, \mathbb{E}^{\pi}\Big[\sum_{h=1}^{H} \bar{c}^i(\bar{s}_h, \Gamma_h)\Big], \\ \bar{V}^{\star}_{\mathrm{alg}} = \sup_{\pi}\big\{\bar{V}^{\pi} :\ \bar{C}^{\pi,i} \leq B_{\mathrm{alg}}/N,\ i = 1, \ldots, N\big\}, \end{gathered} \tag{6.1}$$

where $\mathbb{E}^{\pi}$ is over the draws $\Gamma_h \sim \pi(\cdot \mid \bar{s}_h)$ from $\bar{s}_1 = \varnothing$; the bars distinguish these quantities of $\mathcal{M}$ from V^{π}, $C^{\pi,i}$, and $V^{\star}_{\mathrm{alg}}$ of Section 3; by Proposition A.5, corresponding policies have equal returns and costs, and $\bar{V}^{\star}_{\mathrm{alg}} = V^{\star}_{\mathrm{alg}}$. The *return gap* is $\Delta = \bar{V}^{\star}_{\mathcal{F}} - \bar{V}^{\star}_{\mathcal{H}}$, the difference between the two protocols' optimal returns in Eq. (6.1).

For learning, we replace the learned selections, the action policy, and PPO by a primal–dual learner with a guarantee, which gives HiComm-PD and Flat-PD. In episode $k = 1, \ldots, K$, the agents execute a policy $\pi_{\mathrm{alg},k}$ of $\mathcal{M}$ as above and observe the rewards and the step costs of every agent; between episodes, the method of Stradi et al. (2025), with one dual multiplier for each agent's budget, updates the policy. The learner knows $\overline{\mathcal{A}}_1, \ldots, \overline{\mathcal{A}}_H$, $\bar{A}$, and $\bar{S}$.

Theorem 6.1 (Reward and communication under either protocol)**.** Run HiComm-PD and Flat-PD with the same K and $\delta \in (0,1)$. There is $\mathrm{Err}_K = \widetilde{O}\big(NH^3\bar{S}\sqrt{\bar{A}K} + NH^5\big)$ ($\widetilde{O}$ omits log factors), depending only on $N, H, \bar{S}, \bar{A}, K, \delta$, such that with probability at least $1-\delta$, simultaneously for both protocols and every agent i,

$$\begin{cases} \sum_{k=1}^{K} \bar{V}^{\pi_{\mathcal{F},k}} \geq K\,\bar{V}^{\star}_{\mathcal{F}} - \mathrm{Err}_K/H, & \sum_{k=1}^{K} \bar{C}^{\pi_{\mathcal{F},k},i} \leq \frac{\alpha_{\mathcal{F}}}{N}\big(KH + \mathrm{Err}_K\big), \\ \sum_{k=1}^{K} \bar{V}^{\pi_{\mathcal{H},k}} \geq K\big(\bar{V}^{\star}_{\mathcal{F}} - \Delta\big) - \mathrm{Err}_K/H, & \sum_{k=1}^{K} \bar{C}^{\pi_{\mathcal{H},k},i} \leq \frac{\alpha_{\mathcal{H}}}{N}\big(KH + \mathrm{Err}_K\big). \end{cases} \tag{6.2}$$

Remark 6.2 (What the theorem says)**.** Divided by K, the left sides of Eq. (6.2) are $\bar{V}^{\bar{\pi}_{\mathrm{alg}}}$ and $\bar{C}^{\bar{\pi}_{\mathrm{alg}},i}$ of the learner's output $\bar{\pi}_{\mathrm{alg}}$, which draws k uniformly from $\{1, \ldots, K\}$ with shared randomness and then follows $\pi_{\mathrm{alg},k}$. (i) Both learners have the same learning-error bound $\mathrm{Err}_K/(HK) = \widetilde{O}(K^{-1/2})$, so their reward guarantees differ only by Δ, which does not shrink with K. (ii) Since $\bar{c}^i \leq 1$ and $B_{\mathrm{alg}} = H\alpha_{\mathrm{alg}}$, every policy π of $\mathcal{M}$ satisfies $\bar{C}^{\pi,i} \leq B_{\mathrm{alg}}/N$, so the cost bounds hold trivially; they compare the protocols through their budgets, and HiComm's budget is $1/(N-1)$ of flat communication's. (iii) Since $\bar{S}$ and $\bar{A}$ grow doubly exponentially in H, the bounds compare the protocols but do not predict how many episodes the benchmarks need.

7 Conclusion and Limitations

We formalize cooperative tasks whose entities are organized into publicly indexed groups as the HI-MDP and propose HiComm, in which each receiver asks one teammate, chosen by its position in the hierarchy, and receives the teammate's observation of one entry. In ten IPPO and MAPPO configurations on the three benchmarks, HiComm closes 52%–86% of the reward gap between PartialObs and FullObs, exceeds every learned communication baseline in nine, and communicates less than all of them in all ten. HiComm-PD has the same learning-error bound as Flat-PD, and the two reward guarantees differ only by the return gap Δ (Theorem 6.1), at $1/(N-1)$ of flat communication's budget.

Limitations remain. First, HiComm requires the environment to provide the groups with their entities and agents. Second, Theorem 6.1 holds for HiComm-PD on finite models (Assumption A.7), whereas the experiments train neural policies with PPO, which, to our knowledge, no existing guarantee for constrained MDPs covers.

Reproducibility statement

Appendix A states the message encodings and the communication-cost accounting and gives the complete proofs of the theoretical statements. Appendix B records the implementation, the training objective, and the hyperparameters; the benchmark configurations, fixed opponents, the training seeds (three per configuration), the per-benchmark evaluation averaging, and all result tables; and the ablations. Appendix B.8 lists the compute and software versions and the planned release of the implementation, configuration files, and trained checkpoints after review.

Ethics statement

The study uses public simulators without human participants, personal data, or operational deployment. CC4 models offensive actions only inside its underlying CybORG simulator, not on real networks. Because the CC4 agents act as attackers, the work has a dual-use aspect. The learned policies act on the abstract observation and action spaces of the simulator and cannot be applied to real networks without substantial additional engineering; the paper contributes a communication protocol rather than attack techniques, and any release of trained CC4 checkpoints will follow the terms of the CAGE Challenge 4 repository.

AI use statement

In this work, we used generative AI tools to polish the writing of the main text and of the theoretical analysis, to check the consistency of notation and cross-references, and to format references. All methods, theoretical results and proofs, experiments, and reported numbers were produced and checked by the authors. We take responsibility for the final content of this work, including text, claims or artifacts produced with the aid of generative AI.

References

Eitan Altman. *Constrained Markov Decision Processes*. Stochastic Modeling. Chapman & Hall/CRC, 1999.

Craig Boutilier, Richard Dearden, and Moisés Goldszmidt. Stochastic dynamic programming with factored representations. *Artificial Intelligence*, 121(1–2):49–107, 2000. doi: 10.1016/S0004-3702(00)00033-3.

Abhishek Das, Théophile Gervet, Joshua Romoff, Dhruv Batra, Devi Parikh, Mike Rabbat, and Joelle Pineau. TarMAC: Targeted multi-agent communication. In *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pp. 1538–1546. PMLR, 2019.

Ziluo Ding, Tiejun Huang, and Zongqing Lu. Learning individually inferred communication for multi-agent cooperation. In *Advances in Neural Information Processing Systems*, volume 33, pp. 22069–22079, 2020.

Ziluo Ding, Zeyuan Liu, Zhirui Fang, Kefan Su, Liwen Zhu, and Zongqing Lu. Multi-agent coordination via multi-level communication. In *Advances in Neural Information Processing Systems*, volume 37, pp. 118513–118539, 2024.

Carlos Diuk, Andre Cohen, and Michael L. Littman. An object-oriented representation for efficient reinforcement learning. In *Proceedings of the 25th International Conference on Machine Learning*, pp. 240–247, 2008. doi: 10.1145/1390156.1390187.

Benjamin Ellis, Jonathan Cook, Skander Moalla, Mikayel Samvelyan, Mingfei Sun, Anuj Mahajan, Jakob Foerster, and Shimon Whiteson. SMACv2: An improved benchmark for cooperative multi-agent reinforcement learning. In *Advances in Neural Information Processing Systems*, volume 36, pp. 37567–37593, 2023.

Jakob N. Foerster, Yannis M. Assael, Nando de Freitas, and Shimon Whiteson. Learning to communicate with deep multi-agent reinforcement learning. In *Advances in Neural Information Processing Systems*, volume 29, 2016.

Claudia V. Goldman and Shlomo Zilberstein. Decentralized control of cooperative systems: Categorization and complexity analysis. *Journal of Artificial Intelligence Research*, 22:143–174, 2004. doi: 10.1613/jair.1427.

Serhii Havrylov and Ivan Titov. Emergence of language with multi-agent games: Learning to communicate with sequences of symbols. *Advances in neural information processing systems*, 30, 2017.

Shengchao Hu, Li Shen, Ya Zhang, and Dacheng Tao. Learning multi-agent communication from graph modeling perspective. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=Qox9rO0kN0.

Siyi Hu, Fengda Zhu, Xiaojun Chang, and Xiaodan Liang. UPDeT: Universal multi-agent RL via policy decoupling with transformers. In *International Conference on Learning Representations*, 2021.

Yue Hu, Shaoheng Fang, Zixing Lei, Yiqi Zhong, and Siheng Chen. Where2comm: Communication-efficient collaborative perception via spatial confidence maps. In *Advances in Neural Information Processing Systems*, volume 35, pp. 4874–4886, 2022. URL https://proceedings.neurips.cc/paper_files/paper/2022/hash/1f5c5cd01b864d53cc5fa0a3472e152e-Abstract-Conference.html.

Shariq Iqbal, Christian A. Schroeder de Witt, Bei Peng, Wendelin Böhmer, Shimon Whiteson, and Fei Sha. Randomized entity-wise factorization for multi-agent reinforcement learning. In *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pp. 4596–4606. PMLR, 2021.

Eric Jang, Shixiang Gu, and Ben Poole. Categorical reparameterization with Gumbel-Softmax. In *International Conference on Learning Representations*, 2017. URL https://openreview.net/forum?id=rkE3y85ee.

Jiechuan Jiang and Zongqing Lu. Learning attentional communication for multi-agent cooperation. *Advances in neural information processing systems*, 31, 2018.

Hsu Kao and Vijay Subramanian. Common information based approximate state representations in multi-agent reinforcement learning. In *Proceedings of the 25th International Conference on Artificial Intelligence and Statistics*, volume 151 of *Proceedings of Machine Learning Research*, pp. 6947–6967. PMLR, 2022. URL https://proceedings.mlr.press/v151/kao22a.html.

Mitchell Kiely, Metin Ahiskali, Etienne Borde, Benjamin Bowman, David Bowman, Dirk Van Bruggen, K. C. Cowan, Prithviraj Dasgupta, Erich Devendorf, Ben Edwards, Alex Fitts, Sunny Fugate, Ryan Gabrys, Wayne Gould, H. Howie Huang, Jules Jacobs, Ryan Kerr, Isaiah J. King, Li Li, Luis Martinez, Christopher Moir, Craig Murphy, Olivia Naish, Claire Owens, Miranda Purchase, Ahmad Ridley, Adrian Taylor, Sara Farmer, William John Valentine, and Yiyi Zhang. CAGE challenge 4: A scalable multi-agent reinforcement learning gym for autonomous cyber defence. *AI Magazine*, 46(3):e70021, 2025. doi: 10.1002/aaai.70021.

Daewoo Kim, Sangwoo Moon, David Hostallero, Wan Ju Kang, Taeyoung Lee, Kyunghwan Son, and Yung Yi. Learning to schedule communication in multi-agent reinforcement learning. In *International Conference on Learning Representations*, 2019.

Karol Kurach, Anton Raichuk, Piotr Stańczyk, Michał Zając, Olivier Bachem, Lasse Espeholt, Carlos Riquelme, Damien Vincent, Marcin Michalski, Olivier Bousquet, and Sylvain Gelly. Google research football: A novel reinforcement learning environment. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pp. 4501–4510, 2020. doi: 10.1609/aaai.v34i04.5878.

Angeliki Lazaridou, Alexander Peysakhovich, and Marco Baroni. Multi-agent cooperation and the emergence of (natural) language. In *International Conference on Learning Representations*, 2017. URL https://openreview.net/forum?id=Hk8N3Sclg.

Xinran Li and Jun Zhang. Context-aware communication for multi-agent reinforcement learning. In *Proceedings of the 23rd International Conference on Autonomous Agents and Multiagent Systems*, pp. 1156–1164, 2024.

Xinran Li, Xiaolu Wang, Chenjia Bai, and Jun Zhang. Exponential topology-enabled scalable communication in multi-agent reinforcement learning. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=CL3U0GxFRD.

Zeyang Liu, Lipeng Wan, Xue Sui, Zhuoran Chen, Kewu Sun, and Xuguang Lan. Deep hierarchical communication graph in multi-agent reinforcement learning. In *Proceedings of the Thirty-Second International Joint Conference on Artificial Intelligence*, pp. 208–216, 2023. doi: 10.24963/ijcai.2023/24.

Songtao Lu, Kaiqing Zhang, Tianyi Chen, Tamer Başar, and Lior Horesh. Decentralized policy gradient descent ascent for safe multi-agent reinforcement learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pp. 8767–8775, 2021.

Haipeng Luo, Chen-Yu Wei, and Chung-Wei Lee. Policy optimization in adversarial MDPs: Improved exploration via dilated bonuses. In *Advances in Neural Information Processing Systems*, volume 34, pp. 22931–22942, 2021.

Ashutosh Nayyar, Aditya Mahajan, and Demosthenis Teneketzis. Decentralized stochastic control with partial history sharing: A common information approach. *IEEE Transactions on Automatic Control*, 58(7):1644–1658, 2013. doi: 10.1109/TAC.2013.2239000.

Mitja Nikolaus. Emergent communication with conversational repair. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=Sy8upuD6Bw.

Yaru Niu, Rohan Paleja, and Matthew Gombolay. Multi-agent graph-attention communication and teaming. In *Proceedings of the 20th International Conference on Autonomous Agents and Multi-Agent Systems*, pp. 964–973, 2021.

Frans A. Oliehoek and Christopher Amato. *A Concise Introduction to Decentralized POMDPs*. SpringerBriefs in Intelligent Systems. Springer, 2016. doi: 10.1007/978-3-319-28929-8.

David V. Pynadath and Milind Tambe. The communicative multiagent team decision problem: Analyzing teamwork theories and models. *Journal of Artificial Intelligence Research*, 16:389–423, 2002. doi: 10.1613/jair.1024.

Tabish Rashid, Mikayel Samvelyan, Christian Schroeder de Witt, Gregory Farquhar, Jakob Foerster, and Shimon Whiteson. Monotonic value function factorisation for deep multi-agent reinforcement learning. *Journal of Machine Learning Research*, 21(178):1–51, 2020.

Mikayel Samvelyan, Tabish Rashid, Christian Schroeder de Witt, Gregory Farquhar, Nantas Nardelli, Tim GJ Rudner, Chia-Man Hung, Philip HS Torr, Jakob Foerster, and Shimon Whiteson. The starcraft multi-agent challenge. In *Proceedings of the 18th International Conference on Autonomous Agents and MultiAgent Systems*, pp. 2186–2188, 2019.

Christian Schroeder de Witt, Tarun Gupta, Denys Makoviichuk, Viktor Makoviychuk, Philip H. S. Torr, Mingfei Sun, and Shimon Whiteson. Is independent learning all you need in the StarCraft multi-agent challenge? *arXiv preprint arXiv:2011.09533*, 2020.

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017.

Aditya Vikram Singh, Ethan Rathbun, Emma Graham, Lisa Oakley, Simona Boboila, Peter Chin, and Alina Oprea. Hierarchical multi-agent reinforcement learning for cyber network defense (extended abstract). In *Proceedings of the 24th International Conference on Autonomous Agents and Multiagent Systems*, pp. 2747–2749. IFAAMAS, 2025.

Amanpreet Singh, Tushar Jain, and Sainbayar Sukhbaatar. Learning when to communicate at scale in multiagent cooperative and competitive tasks. In *International Conference on Learning Representations*, 2019.

Yan Song, He Jiang, Haifeng Zhang, Zheng Tian, Weinan Zhang, and Jun Wang. Boosting studies of multi-agent reinforcement learning on Google research football environment: The past, present, and future. In *Proceedings of the 23rd International Conference on Autonomous Agents and Multiagent Systems*, pp. 1772–1781. IFAAMAS, 2024.

Francesco Emanuele Stradi, Matteo Castiglioni, Alberto Marchesi, and Nicola Gatti. Optimal strong regret and violation in constrained MDPs via policy optimization. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://proceedings.iclr.cc/paper_files/paper/2025/hash/8105a80cea247fa6f8441a8069b65f2a-Abstract-Conference.html.

Jayakumar Subramanian, Amit Sinha, Raihan Seraj, and Aditya Mahajan. Approximate information state for approximate planning and reinforcement learning in partially observed systems. *Journal of Machine Learning Research*, 23(12):1–83, 2022.

Sainbayar Sukhbaatar, Arthur Szlam, and Rob Fergus. Learning multiagent communication with backpropagation. In *Advances in Neural Information Processing Systems*, volume 29, 2016.

Chuxiong Sun, Zehua Zang, Jiabao Li, Jiangmeng Li, Xiao Xu, Rui Wang, and Changwen Zheng. T2MAC: Targeted and trusted multi-agent communication through selective engagement and evidence-driven integration. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pp. 15154–15163, 2024a. doi: 10.1609/aaai.v38i13.29438.

Qingshuang Sun, Denis Steckelmacher, Yuan Yao, Ann Nowé, and Raphaël Avalos. Dynamic size message scheduling for multi-agent communication under limited bandwidth. *IEEE Transactions on Mobile Computing*, 23(12):15080–15097, 2024b. doi: 10.1109/TMC.2024.3452986.

Richard S. Sutton, Doina Precup, and Satinder Singh. Between MDPs and semi-MDPs: A framework for temporal abstraction in reinforcement learning. *Artificial Intelligence*, 112(1–2):181–211, 1999. doi: 10.1016/S0004-3702(99)00052-1.

Tonghan Wang, Jianhao Wang, Chongyi Zheng, and Chongjie Zhang. Learning nearly decomposable value functions via communication minimization. In *International Conference on Learning Representations*, 2020. URL https://openreview.net/forum?id=HJx-3grYDB.

Chao Yu, Akash Velu, Eugene Vinitsky, Jiaxuan Gao, Yu Wang, Alexandre Bayen, and Yi Wu. The surprising effectiveness of PPO in cooperative multi-agent games. In *Advances in Neural Information Processing Systems*, volume 35, pp. 24611–24624, 2022.

Vinicius Zambaldi, David Raposo, Adam Santoro, Victor Bapst, Yujia Li, Igor Babuschkin, Karl Tuyls, David Reichert, Timothy Lillicrap, Edward Lockhart, Murray Shanahan, Victoria Langston, Razvan Pascanu, Matthew Botvinick, Oriol Vinyals, and Peter Battaglia. Deep reinforcement learning with relational inductive biases. In *International Conference on Learning Representations*, 2019.

Sai Qian Zhang, Qi Zhang, and Jieyu Lin. Efficient communication in multi-agent reinforcement learning via variance based control. In *Advances in Neural Information Processing Systems*, volume 32, pp. 3230–3239, 2019.

Sai Qian Zhang, Qi Zhang, and Jieyu Lin. Succinct and robust multi-agent communication with temporal message control. *Advances in Neural Information Processing Systems*, 33:17271–17282, 2020.

APPENDIX

Appendix A gives the formal statements and proofs of Section 6, together with the protocol conventions and the communication-cost accounting they use. Appendix B gives the implementation and training details of Section 4, the experimental setup of Section 5, and all results, including the CC4 ablations.

A Theoretical Details and Proofs **15**

A.1 Protocol Conventions and Communication Cost 15

A.2 The Step-Policy MDP 16

A.3 The Return Gap 20

A.4 HICOMM-PD, FLAT-PD, and the Learning Guarantee 20

B Experimental Details and Additional Results **26**

B.1 Implementation and Training Details 26

B.2 Evaluation Protocol and Baselines 28

B.3 CAGE Challenge 4 (CC4) 30

B.4 SMACv2 31

B.5 Google Research Football (GRF) 33

B.6 CC4 Ablation Variants 34

B.7 CC4 Further Tests 34

B.8 Compute, Software, and Availability 35

A THEORETICAL DETAILS AND PROOFS

This appendix gives the formal definitions, statements, and proofs behind Section 6, and Table 6 maps each object and result of Section 6 to its formal version and to the part of the appendix that proves it. Appendix A.1 fixes the protocol conventions and the communication-cost accounting that all proofs use. Appendix A.2 explains what the states and actions of the step-policy MDP $\mathcal{M}$ of Section 6 contain, defines $\mathcal{M}$ and its values formally, and shows that it represents the policy class Π_{alg}, so that its optimal value $\bar{V}^{\star}_{\mathrm{alg}}$ equals the optimum $V^{\star}_{\mathrm{alg}}$ of Eq. (3.2). Appendix A.3 defines the return gap Δ on the step-policy MDPs, and Appendix A.4 states the finite-model assumption, bounds $\bar{S}$ and $\bar{A}$, specifies HICOMM-PD and FLAT-PD, and proves Theorem 6.1. Appendices A.3 and A.4 work in $\mathcal{M}$.

Table 6: Where the objects and results used in Section 6 are defined, stated, and proved.

Section 6	Formal statement	Appendix
$V^{\star}_{\mathrm{alg}}$, Eq. (3.2) (Section 3.3)	Eq. (A.1) and the multi-agent constrained MDP	A.1
$\mathcal{M}$ and Eq. (6.1)	Definition A.2, Proposition A.5	A.2
Return gap Δ	Definition A.6	A.3
HICOMM-PD, FLAT-PD	Algorithms 2 and 3	A.4.2
Finite model	Assumption A.7, Proposition A.8	A.4.1
Theorem 6.1	Theorem 6.1, via Lemmas A.9 and A.10	A.4.3–A.4.4

A.1 PROTOCOL CONVENTIONS AND COMMUNICATION COST

Timing and shared randomness. Within a step, all requests precede all replies, which precede all actions, so a reply cannot be forwarded within the step, and the current step's messages enter a rule only as its explicit arguments; $I^i_1 = (o^i_1, M^i_1)$. Agents may use shared random draws that are independent of the environment and known to every agent, whereas a message is seen only by the agents that send or receive it. Accordingly, Π_{alg} consists of the policies whose requests depend only on I^i_h, whose replies depend only on the sender's history and the received request, and whose actions depend only on (I^i_h, m^i_h), each also on the shared draws.

HICOMM rules. As in Section 4.1, the request rule of receiver i maps I^i_h to silence, written $\widehat{g}^i_h = \perp$, or to a request (q^i_h, j^i_h), a query delivered to one teammate $j^i_h \neq i$. Section 4.2 selects a group $\widehat{g}^i_h \in \widehat{\mathcal{G}}^i$ and then the sender $j^i_h = \widehat{e}^i_h \in \widehat{\mathcal{E}}_{\widehat{g}^i_h} \setminus \{i\}$. Since every agent belongs to a group (Definition 3.1), the request rule can select every teammate $j^i_h \neq i$; the results below use only this property. A teammate may receive several requests in a step and replies to each. Sender j has one reply rule for each receiver i, which maps (I^j_h, q^i_h) to the raw-feature reply $\big((g,e), o^j_h[g,e]\big)$ of Section 4.1 for one entry $(g,e) \in M^j_h$, or to the null reply if $M^j_h = \varnothing$. Receiver i's replies $m^i_h = (m_{h,j\to i})_{j\neq i}$ in Eq. (3.1) hold the reply of j^i_h in slot j^i_h and $\perp$ in every other slot; all slots hold $\perp$ if i is silent.

Encodings. The widths b_q, b_r of Section 3.3 count message bits only; headers, channel coding, and retransmissions are not counted. For fixed N, $\mathcal{G}$, and $\mathcal{E}_g$ the widths are independent of H and of the number of learning episodes; across instances the sender field of a request, which encodes j^i_h, grows as $\lceil \log_2(N-1) \rceil$ bits. A HICOMM request encodes (q^i_h, j^i_h) in b_q bits; when a query encodes an entry index, this index is part of q^i_h and uses at most $\lceil \log_2 \sum_{g\in\mathcal{G}} |\mathcal{E}_g| \rceil$ bits within b_q. A flat context is any b_q-bit message (Sections 3.2 and 3.3), so it can carry this request. A HICOMM reply encodes $(g^i_h, e^i_h, o^{j^i_h}_h[g^i_h, e^i_h])$ at a fixed numerical precision, or the reserved null codeword, in b_r bits; a flat reply is any b_r-bit message, so it can carry the same reply. An active HICOMM request, one that is not silence, receives a transmitted reply, which may be the null reply.

Step costs. At a fixed step h, the step cost c^i_h of agent i (Section 3.3) counts the bits delivered in the exchange that receiver i starts, each delivered copy separately. Under HICOMM and under flat

communication it is

$$c_h^{\mathcal{H},i} = (b_q + b_r)\,\mathbf{1}\{\widehat{g}_h^i \neq \perp\},$$
$$c_h^{\mathcal{F},i} = (N-1)b_q\,\mathbf{1}\{q_h^i \neq \perp\} + b_r\big|\{j \neq i :\ j \text{ transmits a reply to } i\}\big|, \tag{A.1}$$

and the team step costs are $c_h^{\mathrm{alg}} = \sum_{i=1}^N c_h^{\mathrm{alg},i}$. The reply count includes transmitted null replies; only active contexts can receive replies. Each HICOMM request is delivered once, whereas each active flat context is delivered to $N-1$ other agents. Every delivered message belongs to exactly one exchange, so the per-agent costs split the team cost without double counting.

Capacities. Equation (A.1) gives $0 \le c_h^{\mathcal{H},i} \le b_q + b_r$ and $0 \le c_h^{\mathcal{F},i} \le (N-1)(b_q + b_r)$, with equality when agent i's request is active and, in flat communication, every teammate replies. When every permitted message is active, both protocols have N initiating requests, delivering N request copies in HICOMM and $N(N-1)$ in flat communication, and the reply count equals N or $N(N-1)$. The team step costs are then exactly $\alpha_{\mathcal{H}} = N(b_q + b_r)$ and $\alpha_{\mathcal{F}} = N(N-1)(b_q + b_r)$, so each agent's maximal step cost is α_{alg}/N, and $\alpha_{\mathcal{F}}/\alpha_{\mathcal{H}} = N-1$. Counted in messages, these all-active steps deliver $2N$ messages in HICOMM and $2N(N-1)$ in flat communication, that is, $\Theta(N)$ and $\Theta(N^2)$.

The multi-agent constrained MDP. Problem (3.2) with the added per-agent constraints $C^{\pi,i} \le B_{\mathrm{alg}}/N$, $i = 1, \dots, N$, follows the networked multi-agent constrained MDP (MA-CMDP) of Lu et al. (2021, Definition 1 and Eq. (1)): the team maximizes a shared return, and each agent i has its own cost c_h^i and its own constraint. The constraints leave the optimum $V_{\mathrm{alg}}^\star$ unchanged: by the capacities above, $c_h^{\mathrm{alg},i} \le \alpha_{\mathrm{alg}}/N$, so $C^{\pi,i} \le H\alpha_{\mathrm{alg}}/N = B_{\mathrm{alg}}/N$ for every $\pi \in \Pi_{\mathrm{alg}}$. Three differences come from our setting. (i) The horizon is finite, H, instead of discounted. (ii) The state is not observed: each agent observes o_h^i and M_h^i, and the communication protocol alg, rather than a consensus network, fixes what the agents may exchange; the policy class Π_{alg} of Section 3.2 therefore replaces the product of local state-feedback policies. (iii) The costs are communication costs, which the protocol determines. The analysis is carried out on the step-policy MDP $\mathcal{M}$ (Appendix A.2), whose actions are step policies; Proposition A.5 shows that it represents every policy of Π_{alg}.

A.2 THE STEP-POLICY MDP

This subsection concerns the step-policy MDP $\mathcal{M}$ of Section 6. Appendix A.2.1 explains what its states and actions contain and how they differ from those of the HI-MDP, Appendix A.2.2 defines it formally, and Appendix A.2.3 defines its values and relates it to the policy class Π_{alg}. Fix a protocol $\mathrm{alg} \in \{\mathcal{H}, \mathcal{F}\}$; the step-policy sets, prefixes, policies, and tables below are defined for this protocol. Throughout, a step policy is deterministic.

A.2.1 STATES AND ACTIONS OF THE STEP-POLICY MDP

Role of the step-policy MDP. In the HI-MDP, the decision at step h is split among the N agents, and each agent sees only its own local history I_h^i and the messages it receives. The analysis needs quantities that all agents know and that fix the joint behavior at each step. $\mathcal{M}$ provides them, following the common-information approach of Nayyar et al. (2013): its state, the prefix, is information that all agents share, and its action, a step policy, contains one rule for each agent that maps the agent's private information to its decision. Table 7 compares the two MDPs.

The action: what a step policy contains. A step policy is a table of decisions rather than one decision. For every agent, it contains three rules for step h. Under HICOMM:

- the request rule of receiver i says, for every history that agent i may have at step h, whether i stays silent or which teammate j_h^i it asks and with which query q_h^i;
- the reply rule of sender j to receiver i says, for every history of j and every query from i, which entry among those that j observes it returns with its features, or the null reply when j observes no entry;

Table 7: The HI-MDP and the step-policy MDP $\mathcal{M}$ of protocol alg at step h.

	HI-MDP (Section 3.1)	Step-policy MDP $\mathcal{M}$ (Section 6)
State	environment state $s_h \in \mathcal{S}$; no agent observes it	prefix $\bar{s}_h = (\Gamma_1, \ldots, \Gamma_{h-1}) \in \overline{\mathcal{S}}_h$; every agent knows it
Action	joint action $\mathbf{a}_h \in \mathcal{A}$; agent i chooses a_h^i after the communication round	step policy $\Gamma_h \in \overline{\mathcal{A}}_h$: the request, reply, and action rules of all agents for every history they may have
Transition	$s_{h+1} \sim P(\cdot \mid s_h, \mathbf{a}_h)$, unknown to the learner	$\bar{s}_{h+1} = (\bar{s}_h, \Gamma_h)$, deterministic and known
Reward and costs	$r(s_h, \mathbf{a}_h)$ and the step costs c_h^i in bits	means $\bar{r}(\bar{s}_h, \Gamma_h)$ and $\bar{c}^i(\bar{s}_h, \Gamma_h)$, unknown to the learner
Policy	a policy in Π_{alg}: rules of every agent that read its local history, and shared draws	a distribution $\pi(\cdot \mid \bar{s}_h)$ on $\overline{\mathcal{A}}_h$ for every prefix

- the action rule of agent i says, for every history of i and every reply that i may receive, which action a_h^i it takes.

Under flat communication, the request rule chooses a b_q-bit context or silence, the reply rule of every teammate chooses a b_r-bit reply or $\perp$, and the action rule reads all $N-1$ reply slots. Since every rule is deterministic, all randomness of a policy of $\mathcal{M}$ lies in the draw of Γ_h, which uses only the shared randomness and the prefix, and hence no private information. Each agent uses its private information only when it applies its rules to the history it actually has, as in Algorithms 2 and 3. Appendix A.2.2 makes "history" precise: the rules read the part of the local history that the agent did not choose, which loses no generality (Lemma A.4). Because a step policy assigns an output to every possible history, $\overline{\mathcal{A}}_h$ is large; Proposition A.8 bounds $|\overline{\mathcal{A}}_h|$ and $|\overline{\mathcal{S}}|$, and the bounds grow doubly exponentially in H.

The state: why prefixes suffice. Every agent knows the prefix $\bar{s}_h$, because the step policies in it were drawn with shared randomness. The prefix contains neither the environment state nor any agent's history, but it fixes their joint law. Starting from $s_1 \sim b_0$, the rules of $\Gamma_1, \ldots, \Gamma_{h-1}$ determine every message and action as a function of the realized initial state and transitions, so the prefix determines the law of s_h and of all local histories at step h. For any step policy $\Gamma \in \overline{\mathcal{A}}_h$, the expected reward and step costs at step h when the agents apply Γ are therefore fixed numbers, the mean reward $\bar{r}(\bar{s}_h, \Gamma)$ and the mean costs $\bar{c}^i(\bar{s}_h, \Gamma)$. The uncertainty about s_h is averaged into these means instead of being part of the state, and the transition, which only appends Γ_h, is deterministic. The randomness of the environment remains in the realized rewards and costs, which the learner observes after each episode; the means $\bar{r}$ and $\bar{c}^i$ are unknown to it.

An example. Let $N = 2$ under HiComm, and suppose that agent 2 observes an entry (g, e) at every step and agent 1 never does. In one step policy Γ, the request rule of agent 1 always asks agent 2 with a fixed query, and that of agent 2 is always silent; the reply rule of agent 2 to agent 1 returns (g, e) and its features; and the action rule of each agent maps its observation and the reply, if any, to an action by a fixed table. The policy of $\mathcal{M}$ that plays Γ at every step visits the prefixes $\varnothing, (\Gamma), (\Gamma, \Gamma), \ldots$ and uses no randomness. At every step it spends $b_q + b_r$ bits in the exchange of agent 1 and none in that of agent 2.

Relation to the policy class. A policy in Π_{alg} may randomize privately and reads local histories, whereas a policy of $\mathcal{M}$ randomizes only through the shared draw of Γ_h. Proposition A.5 shows that this loses nothing: every policy in Π_{alg} has the trajectory law (the joint law of the states, observations, messages, and actions of an episode) of the execution of a policy of $\mathcal{M}$, and conversely. Hence the value, the costs, and the optimum of Eqs. (A.2) and (A.3) are those of the MA-CMDP of Appendix A.1, and $\mathcal{M}$ is a constrained MDP with one constraint per agent, to which the primal–dual method of Stradi et al. (2025) applies (Appendix A.4).

A.2.2 Formal Definition of the Step-Policy MDP

Definition A.1 (Step policies and prefixes)**.** The *received history* J_h^i of agent i at step h is the part of its local history I_h^i that it did not choose: its observations and masks through step h and the messages it received at earlier steps. A deterministic step policy (Section 6) consists of the deterministic request, reply, and action rules of all agents for one step (Nayyar et al., 2013), each of which reads the agent's received history in place of its local history; it is denoted by Γ. Lemma A.4 shows that this restriction loses no generality. The set of all deterministic step policies at step h whose outputs protocol alg admits (listed after Definition A.2) is $\overline{\mathcal{A}}_h$. The prefix at step h is $\bar{s}_h = (\Gamma_1, \dots, \Gamma_{h-1})$, with $\Gamma_{h'} \in \overline{\mathcal{A}}_{h'}$ for $h' = 1, \dots, h-1$; the set of all such prefixes is $\overline{\mathcal{S}}_h$. Thus the prefix at step 1 is the empty sequence, $\bar{s}_1 = \varnothing$, and $\bar{s}_{h+1} = (\bar{s}_h, \Gamma_h)$. Policies and values on the step-policy MDP below are functions of the prefix; a prefix in $\overline{\mathcal{S}}_h$ has length $h-1$, so it determines the step h. The set of all prefixes is $\overline{\mathcal{S}} = \bigcup_{h=1}^{H} \overline{\mathcal{S}}_h$ (Section 6); these sets are disjoint, so $|\overline{\mathcal{S}}| = \sum_{h=1}^{H} |\overline{\mathcal{S}}_h|$.

For $\bar{s}_h = (\Gamma_1, \dots, \Gamma_{h-1}) \in \overline{\mathcal{S}}_h$ and $\Gamma \in \overline{\mathcal{A}}_h$, consider the run of the HI-MDP from $s_1 \sim b_0$ in which the agents apply the rules of $\Gamma_{h'}$ at every step $h' < h$ and the rules of Γ at step h. The expected reward $r(s_h, \mathbf{a}_h)$ of this run is the *mean reward* $\bar{r}(\bar{s}_h, \Gamma) \in [0, 1]$. The expected step cost $c_h^{\mathrm{alg},i}$ of this run, divided by its largest value α_{alg}/N (Appendix A.1), is the *mean cost* $\bar{c}^i(\bar{s}_h, \Gamma) \in [0, 1]$ of agent i. Both are well defined because the prefix and Γ fix the rules of steps $1, \dots, h$, and hence the law of $(s_h, \mathbf{a}_h)$ and of the step costs.

Definition A.2 (Step-policy MDP)**.** The *step-policy MDP* of protocol alg is the constrained MDP $\mathcal{M} = \big(\overline{\mathcal{S}}, (\overline{\mathcal{A}}_h)_{h=1}^{H}, \bar{r}, (\bar{c}^i)_{i=1}^{N}\big)$ with horizon H and the following components.

- *States and actions.* At step h, the state is a prefix $\bar{s}_h \in \overline{\mathcal{S}}_h$ and the action is a step policy $\Gamma_h \in \overline{\mathcal{A}}_h$; the initial state is $\bar{s}_1 = \varnothing$.
- *Transitions.* The transition is deterministic, $\bar{s}_{h+1} = (\bar{s}_h, \Gamma_h)$, and the episode ends after step H.
- *Rewards and costs.* At step h, the reward and the N costs are the reward $r(s_h, \mathbf{a}_h)$ and the rescaled step costs $N c_h^{\mathrm{alg},i}/\alpha_{\mathrm{alg}}$ of the HI-MDP run in which the agents apply the rules of $\Gamma_1, \dots, \Gamma_h$; their conditional means given $(\bar{s}_h, \Gamma_h)$ are $\bar{r}(\bar{s}_h, \Gamma_h)$ and $\bar{c}^i(\bar{s}_h, \Gamma_h)$.
- *Constraints.* Constraint i bounds the expected sum of the i-th cost over the H steps by H; in bits, this is agent i's share B_{alg}/N of the budget (Section 3.3).
- *Policies and values.* A policy π of $\mathcal{M}$ is a distribution $\pi(\cdot \mid \bar{s}_h)$ on $\overline{\mathcal{A}}_h$ for each prefix $\bar{s}_h \in \overline{\mathcal{S}}_h$. Its value, its costs, and the optimal value are

$$\bar{V}^{\pi} = \mathbb{E}^{\pi}\Big[\sum_{h=1}^{H} \bar{r}(\bar{s}_h, \Gamma_h)\Big], \qquad \bar{C}^{\pi,i} = \frac{\alpha_{\mathrm{alg}}}{N}\, \mathbb{E}^{\pi}\Big[\sum_{h=1}^{H} \bar{c}^i(\bar{s}_h, \Gamma_h)\Big], \tag{A.2}$$

$$\bar{V}^{\star}_{\mathrm{alg}} = \sup\big\{\bar{V}^{\pi} :\ \pi \text{ is a policy of } \mathcal{M},\ \bar{C}^{\pi,i} \le B_{\mathrm{alg}}/N \text{ for } i = 1, \dots, N\big\}, \tag{A.3}$$

  where $\mathbb{E}^{\pi}$ is the expectation over the draws $\Gamma_h \sim \pi(\cdot \mid \bar{s}_h)$ from $\bar{s}_1 = \varnothing$.

The team executes a policy π of $\mathcal{M}$ in the HI-MDP by drawing each $\Gamma_h \sim \pi(\cdot \mid \bar{s}_h)$ with shared randomness, and every agent then applies its rules in Γ_h. A policy of Π_{alg} is identified with its representation as a policy of $\mathcal{M}$ (Proposition A.5); the episode-k policy of the learner of protocol alg is $\pi_{\mathrm{alg},k}$ (Section 6), written π_k when alg is fixed.

In both protocols, the request rule of receiver i, the reply rule of sender j to receiver i, and the action rule of agent i in a step policy Γ are denoted by $\Gamma^{i,\mathrm{q}}$, $\Gamma^{j\to i,\mathrm{r}}$, and $\Gamma^{i,\mathrm{a}}$. Their inputs are respectively J_h^i, (J_h^j, q_h^i), and (J_h^i, m_h^i); since J_h^i is part of I_h^i, we also write $\Gamma^{i,\mathrm{q}}(I_h^i)$ for $\Gamma^{i,\mathrm{q}}(J_h^i)$, as in Algorithms 2 and 3, and likewise for the other rules. Agent i's component in Section 6 is $\Gamma^i = \big(\Gamma^{i,\mathrm{q}}, (\Gamma^{i\to j,\mathrm{r}})_{j\neq i}, \Gamma^{i,\mathrm{a}}\big)$. The protocols differ in the outputs.

- HiComm. The request rule returns silence or (q_h^i, j_h^i) with $j_h^i \neq i$, encoded in b_q bits. Only sender j_h^i applies its reply rule to receiver i; the rule selects an entry that the sender observes and returns its index and features, or the null reply if the sender observes no entry (Appendix A.1). The replies m_h^i hold at most one non-$\perp$ slot.

- *Flat communication.* The request rule returns a context $q_h^i \in \{0,1\}^{b_q}$ or silence $\perp$, and an active context is delivered to all $N-1$ teammates. Every teammate $j \neq i$ applies its reply rule to receiver i, which returns a reply in $\{0,1\}^{b_r}$ or $\perp$. The replies m_h^i may hold $N-1$ non-$\perp$ slots.

A.2.3 Values and the Reduction to Step Policies

Definition A.3 (Values on the step-policy MDP)**.** For a table f on prefix–step-policy pairs, extend the value notation of Eq. (A.2) by

$$\bar{V}^{\pi}(f) = \mathbb{E}^{\pi}\left[\sum_{h=1}^{H} f(\bar{s}_h, \Gamma_h)\right]. \tag{A.4}$$

The mean reward $\bar{r}$ and the mean costs $\bar{c}^i$ of Appendix A.2.2, the latter in units of α_{alg}/N (Appendix A.1), are unknown to the learner. By Eq. (A.2), $\bar{V}^{\pi} = \bar{V}^{\pi}(\bar{r})$ and $\bar{C}^{\pi,i} = (\alpha_{\mathrm{alg}}/N)\bar{V}^{\pi}(\bar{c}^i)$, so agent i's constraint $\bar{C}^{\pi,i} \leq B_{\mathrm{alg}}/N$ in Eq. (A.3) reads $\bar{V}^{\pi}(\bar{c}^i) \leq H$, which every π satisfies because $\bar{c}^i \leq 1$.

Lemma A.4 (Received histories suffice)**.** Fix a sequence of deterministic request, reply, and action rules for steps $1, \ldots, H$ whose inputs are local histories, as in Section 3.2. There is a sequence of step policies $\Gamma_1, \ldots, \Gamma_H$, whose rules read received histories, such that on every realization of the initial state and the transitions the two executions send the same messages and take the same actions. The step policy Γ_h depends only on the given rules for steps $1, \ldots, h$.

Proof. Given rules for steps $1, \ldots, h-1$, we show by induction on h that every agent's local history I_h^i is a function φ_h^i of its received history J_h^i, where φ_h^i depends only on those rules. For $h = 1$, $I_1^i = J_1^i = (o_1^i, M_1^i)$. Suppose the claim holds at step h. The local history I_{h+1}^i adds to I_h^i the agent's request at step h, which is the request rule applied to I_h^i; its reply to each teammate j, which is the reply rule applied to (I_h^i, q_h^j), where q_h^j is a request that i received and hence part of J_{h+1}^i; its action, which is the action rule applied to (I_h^i, m_h^i), where m_h^i is part of J_{h+1}^i; the messages it received at step h; and (o_{h+1}^i, M_{h+1}^i). With $I_h^i = \varphi_h^i(J_h^i)$ and J_h^i part of J_{h+1}^i, all of these are functions of J_{h+1}^i, which proves the claim at step $h+1$. Define Γ_h by composing each rule at step h with φ_h^i (for a reply rule, with φ_h^j of the sender). On every realization, the two executions then have the same local histories, messages, and actions at every step, by induction on h. □

Proposition A.5 (Equivalent policy representations)**.** Every policy in Π_{alg} has a representation as a policy of $\mathcal{M}$ whose execution induces the same trajectory law, and the execution of every policy of $\mathcal{M}$ is a policy in Π_{alg}. Consequently, for a policy π of $\mathcal{M}$, the value $\bar{V}^{\pi}$ and the costs $\bar{C}^{\pi,i}$ of Eq. (A.2) are the return and the per-agent costs, as defined in Section 3, of the policy in Π_{alg} that π represents, and the optimal value $\bar{V}^{\star}_{\mathrm{alg}}$ of Eq. (A.3) is the optimum $V^{\star}_{\mathrm{alg}}$ of Eq. (3.2).

Proof of Proposition A.5. Replace each agent's private randomization by shared draws that only that agent's rules read, which leaves the trajectory law unchanged, and fix all shared draws before an episode. At every possible local input, its choices are then deterministic, yielding a distribution over complete sequences of rules with local-history inputs. Lemma A.4 maps each such sequence to a sequence of step policies with the same messages and actions on every realization; pushing the distribution forward through this map gives a distribution over complete sequences of step policies with the same trajectory law. Factoring this distribution as $\pi(\Gamma_h \mid \Gamma_1, \ldots, \Gamma_{h-1})$ produces a policy of $\mathcal{M}$ for each policy in Π_{alg}; probabilities at zero-probability prefixes may be assigned arbitrarily. Conversely, the shared randomness samples step policies from any such distribution, and each agent applies the sampled maps to its allowed local inputs; a rule that reads the received history is a rule of Π_{alg}, since the received history is part of the local history. For a fixed sequence of step policies, this execution and the original policy have the same conditional environment law. Matching sequence distributions therefore preserves the full trajectory law, rewards, per-agent communication costs, and optimum. Randomizing between policies at the episode's start remains within Π_{alg}. For the consequence, fix a policy π of $\mathcal{M}$ and execute it. The draw of Γ_h uses only the shared randomness and the prefix, so, given $\Gamma_1, \ldots, \Gamma_h$, the pair $(s_h, \mathbf{a}_h)$ has the law of the run with these fixed rules that defines $\bar{r}$ and $\bar{c}^i$ in Appendix A.2.2. Hence $\mathbb{E}[r(s_h, \mathbf{a}_h)] = \mathbb{E}^{\pi}[\bar{r}(\bar{s}_h, \Gamma_h)]$ and $\mathbb{E}[c_h^{\mathrm{alg},i}] = (\alpha_{\mathrm{alg}}/N)\,\mathbb{E}^{\pi}[\bar{c}^i(\bar{s}_h, \Gamma_h)]$ at every step h, and summing over h gives the return and the

costs. The two optima coincide because the two policy classes correspond with equal returns and costs and, by Definition A.3, every policy of $\mathcal{M}$ satisfies the constraints of Eq. (A.3). □

All environment randomness of one episode (the initial state and every transition) can be generated from one sequence of independent uniform draws, the episode's random tape; draw a fresh tape independently for each episode. Replaying every prefix and next step policy on that tape defines the joint reward table and the N rescaled communication-cost tables, with means $\bar{r}, \bar{c}^1, \ldots, \bar{c}^N$. Conditional on earlier episodes, prefix selection uses no current environment feedback, so visits are independent of the fresh tape. Table entries within an episode may be correlated; the episode tables are independent draws from the same distribution.

A.3 The Return Gap

Definition A.6 (Return gap)**.** The *return gap* is $\Delta = \bar{V}^\star_{\mathcal{F}} - \bar{V}^\star_{\mathcal{H}}$, where $\bar{V}^\star_{\mathcal{F}}$ and $\bar{V}^\star_{\mathcal{H}}$ are the optimal values, Eq. (A.3), of the step-policy MDPs of flat communication and HiComm (Definition A.2).

The gap enters the analysis only in the last step of the proof of Theorem 6.1, which substitutes $\bar{V}^\star_{\mathcal{H}} = \bar{V}^\star_{\mathcal{F}} - \Delta$.

A.4 HiComm-PD, Flat-PD, and the Learning Guarantee

This subsection states the finite-model assumption, bounds $\bar{S}$ and $\bar{A}$, specifies HiComm-PD and Flat-PD, and proves Theorem 6.1; it works in the step-policy MDP of Appendix A.2. Appendix A.4.1 states Assumption A.7 and proves Proposition A.8. Appendix A.4.2 specifies HiComm-PD (Algorithm 2) and Flat-PD (Algorithm 3), which run the Constrained Primal–Dual Policy Optimization (CPD-PO) algorithm of Stradi et al. (2025) on the step-policy MDPs of HiComm and of flat communication. Appendices A.4.3–A.4.4 prove the theorem: $\mathcal{M}$, which has one constraint per agent, satisfies the assumptions of Stradi et al. (2025); their regret and violation theorems then bound, for each protocol separately, the summed per-episode shortfall of the return and, for every agent, the summed excess of its cost; and the theorem compares the two bounds. Remark 6.2 in Section 6 explains what the theorem says, and the paragraph at the end of this subsection adds technical details.

A.4.1 Finite Model and Size Bounds

The learning analysis uses only Assumption A.7 below and the episodic setting of Section 6. Fix a protocol $\mathrm{alg} \in \{\mathcal{H}, \mathcal{F}\}$; the proof of Theorem 6.1 compares the two protocols only through quantities that carry the tag alg.

Assumption A.7 (Finite model)**.** The state set $\mathcal{S}$ and the action sets $\mathcal{A}^i$ of the HI-MDP are finite; the learner knows the sets $\overline{\mathcal{A}}_1, \ldots, \overline{\mathcal{A}}_H$ and upper bounds $\bar{A} \geq \max_h |\overline{\mathcal{A}}_h|$ and $\bar{S} \geq |\overline{\mathcal{S}}|$, the same for both protocols.

A deterministic step policy at step h assigns a request to every received history, a reply to every pair of a sender's received history and a received request, for each receiver, and an action to every pair of a received history and the received replies; $\bar{S}$ counts prefixes of such step policies. The statements use the same $\bar{S}, \bar{A}$ for both protocols, and the flat sizes are valid for both: under the encodings of Appendix A.1, every HiComm request and reply is a flat context and reply of the same width, so HiComm received histories map injectively into flat ones and every HiComm rule has no more inputs and no more possible outputs than the corresponding flat rule; hence HiComm's sets are no larger. Proposition A.8 below gives explicit values.

Proposition A.8 (Size bounds)**.** Let $\mathcal{S}$ and $\mathcal{A}$ be finite, and let

$$\Lambda = \Big(|\mathcal{S}| \left[(2^{b_q}+1)(2^{b_r}+1)\right]^{N-1}\Big)^{H} \times \Big[N \log_2(2^{b_q}+1) + N(N-1)2^{b_q} \log_2(2^{b_r}+1) + (2^{b_r}+1)^{N-1} \log_2 |\mathcal{A}|\Big].$$

Under either protocol,

$$|\overline{\mathcal{A}}_h| \leq 2^{\Lambda} \ \ (1 \leq h \leq H), \qquad |\overline{\mathcal{S}}| \leq H\, 2^{\Lambda/8}.$$

Besides N and H, both bounds depend on the environment only through $|\mathcal{S}|$ and $|\mathcal{A}|$, and on the protocol only through the message widths b_q and b_r of Section 3.3.

Proof of Proposition A.8. Fix a protocol. By Lemma A.4, the rules of a step policy read received histories (Definition A.1). The proof counts received histories, then the rules of a step policy, then prefixes; the count of rules is the number of prescriptions in the sense of Nayyar et al. (2013), which are our step policies. Write

$$G = |\mathcal{S}| \left[(2^{b_q}+1)(2^{b_r}+1)\right]^{N-1},$$
$$W = N \log_2(2^{b_q}+1) + N(N-1)2^{b_q} \log_2(2^{b_r}+1) + (2^{b_r}+1)^{N-1} \log_2 |\mathcal{A}|,$$

the two factors of Λ in the proposition, so that $\Lambda = G^H W$.

Step 1: received histories. Agent i's observation and mask at a step are a function of s_h (Section 3.1), so together they take at most $|\mathcal{S}|$ values. The received history J_1^i is (o_1^i, M_1^i). The received history J_{h+1}^i extends J_h^i by the request received from each of the $N-1$ teammates at step h (at most $2^{b_q}+1$ values each, $\perp$ included, since a HICOMM request and a flat context are both b_q-bit messages); the reply received from each of the $N-1$ teammates (at most $2^{b_r}+1$ values each, $\perp$ included); and (o_{h+1}^i, M_{h+1}^i) (at most $|\mathcal{S}|$ values). Hence an agent has at most $|\mathcal{S}|G^{h-1} \le G^h$ received histories at step h.

Step 2: step policies. A deterministic step policy at step h consists, for every agent, of one request rule, $N-1$ reply rules, one per receiver, and one action rule. A request rule maps each of at most G^h received histories to at most $2^{b_q}+1$ outputs. A reply rule maps each pair of a received history and one of at most 2^{b_q} transmitted requests to at most $2^{b_r}+1$ outputs ($\perp$ or the null reply included). An action rule of agent i maps each pair of a received history and a value of m_h^i, of which there are at most $(2^{b_r}+1)^{N-1}$, to $\mathcal{A}^i$. Multiplying over the N agents and using $\sum_{i=1}^N \log_2 |\mathcal{A}^i| = \log_2 |\mathcal{A}|$,

$$\log_2 |\overline{\mathcal{A}}_h| \le G^h W \le G^H W = \Lambda,$$

so $|\overline{\mathcal{A}}_h| \le 2^\Lambda$ for every h.

Step 3: prefixes. A prefix in $\overline{\mathcal{S}}_h$ is a sequence of $h-1$ step policies. The widths b_q, b_r are positive integers, so $G \ge 3^{2(N-1)} \ge 9$. By Step 2,

$$\log_2 |\overline{\mathcal{S}}_h| = \sum_{h'=1}^{h-1} \log_2 |\overline{\mathcal{A}}_{h'}| \le W \sum_{h'=1}^{h-1} G^{h'} = W \frac{G^h - G}{G-1} \le \frac{W G^H}{8} = \frac{\Lambda}{8},$$

so $|\overline{\mathcal{S}}| = \sum_{h=1}^H |\overline{\mathcal{S}}_h| \le H\, 2^{\Lambda/8}$. Steps 1–3 use only that every message is a b_q-bit request or a b_r-bit reply, so the bounds hold under both protocols. Hence $\bar{A} = 2^\Lambda$ and $\bar{S} = H\, 2^{\Lambda/8}$ satisfy Assumption A.7. □

The count of rules in Step 2 is exact for flat communication, where every b_q-bit context and every b_r-bit reply is admissible; the slack is in the count of received histories, which ignores which of them are reachable. The bounds grow doubly exponentially in H because a step policy assigns an output to every received history, as the number of deterministic joint policies of a Dec-POMDP does (Oliehoek & Amato, 2016); this growth is unavoidable for history-dependent policies. HICOMM's sets are smaller: a HICOMM receiver gets at most one reply, so m_h^i takes at most $(N-1)2^{b_r}+1$ values instead of $(2^{b_r}+1)^{N-1}$, both in the received history and as an input of the action rule. Theorem 6.1 uses one pair $\bar{S}, \bar{A}$ for both protocols.

A.4.2 THE LEARNERS HICOMM-PD AND FLAT-PD

All requests precede all replies. Resetting an episode initializes each local history. Executing a joint action records the reward and every agent's step communication cost, then advances each local history using only that agent's observations, masks, actions, and sent and received messages. The learner knows α_{alg}, K, and δ in addition to the quantities of Assumption A.7, and it receives the episode's rewards and step costs only after the episode.

HICOMM-PD (Algorithm 2) runs CPD-PO (Algorithm 2 of Stradi et al. (2025)) on the step-policy MDP $\mathcal{M}$ of HICOMM (Definition A.2), which has one constraint per agent. CPD-PO works on

loop-free constrained MDPs, whose states are partitioned into layers that every episode traverses in order (Appendix A.4.3), and takes as input their Slater margin, the largest ρ such that some policy keeps every expected cost at least ρ below its threshold (Lemma A.9(iii)). CPD-PO uses the policy optimization with dilated bonuses of Luo et al. (2021) as its primal algorithm, estimates rewards and costs from the visited pairs, and at each episode sets each dual multiplier to one of two values, 0 or the number of layers plus one divided by the Slater margin, according to an optimistic estimate of the corresponding constraint; we use it exactly as specified by Stradi et al. (2025) and do not restate its updates. Its communication round is that of Algorithm 1: a query to one sender and a reply about one entry that the sender observes. Lines 6–7 of Algorithm 2 carry out the query, the sender selection, and the reply of lines 2–5 of Algorithm 1, with the rules of the sampled step policy in place of the selection function sel. Its action rules read the local history and the reply (Eq. (3.1)) rather than only the written observations, and its policy and update differ. It reads each agent's step communication cost divided by its largest value α_{alg}/N, that is, $c_h^{\mathrm{alg},i}/(\alpha_{\mathrm{alg}}/N) \in [0,1]$, so each agent's largest possible cost, its share B_{alg}/N of the budget, becomes H, and the same algorithm serves both protocols; the policy remains fixed within each episode. Its inputs are the number of layers, the number of constraints and their thresholds, the Slater margin, and a confidence level. We set them to

$$H \text{ layers}, \quad N \text{ constraints with thresholds } H, \quad \text{Slater margin } H, \quad \text{confidence } \delta' = \frac{\delta}{4c_0}, \tag{A.5}$$

where c_0 is the absolute constant in the failure probability $1 - O(\delta')$ of Theorems 1 and 2 of Stradi et al. (2025). With these inputs, the two values of each dual multiplier are 0 and $(H+1)/H$.

Algorithm 2 HiComm-PD. At each step h, lines 6–8 run for every receiver i in parallel.

Input: Step-policy sets $\overline{\mathcal{A}}_1, \dots, \overline{\mathcal{A}}_H$ of HiComm; $\bar{S}, \bar{A}, \alpha_{\mathcal{H}}, K, \delta$; the inputs of Eq. (A.5). The initial distribution, transitions, and rewards are unknown.

1: Initialize CPD-PO (Stradi et al., 2025, Algorithm 2) on $\mathcal{M}$ of HiComm (Definition A.2) with the inputs of Eq. (A.5); $\pi_1 \leftarrow$ its initial policy.
2: **for** episode $k = 1, \dots, K$ **do**
3: Reset the environment and the local histories; $\bar{s}_{k,1} \leftarrow \varnothing$.
4: **for** step $h = 1, \dots, H$ **do**
5: $\Gamma_{k,h} \sim \pi_k(\cdot \mid \bar{s}_{k,h})$ by shared randomness. step policy
6: $(q_h^i, j_h^i) \leftarrow \Gamma_{k,h}^{i,\mathrm{q}}(I_h^i)$, or silence, $\widehat{g}_h^i = \perp$; if $\widehat{g}_h^i = \perp$, go to line 8. query and select sender
7: Sender $j = j_h^i$: $\big((g_h^i, e_h^i), o_h^j[g_h^i, e_h^i]\big) \leftarrow \Gamma_{k,h}^{j \to i,\mathrm{r}}(I_h^j, q_h^i)$ with $(g_h^i, e_h^i) \in M_h^j$, or null. reply
8: $a_h^i \leftarrow \Gamma_{k,h}^{i,\mathrm{a}}(I_h^i, m_h^i)$, with the replies m_h^i of Eq. (3.1). act
9: Execute $\mathbf{a}_h$; record the reward $r_{k,h}$ and the step costs $c_{k,h}^i$ of Eq. (A.1); advance the local histories; $\bar{s}_{k,h+1} \leftarrow (\bar{s}_{k,h}, \Gamma_{k,h})$.
10: **end for**
11: Pass the visited pairs $(\bar{s}_{k,h}, \Gamma_{k,h})$ with rewards $r_{k,h}$ and rescaled costs $N c_{k,h}^i/\alpha_{\mathcal{H}}$, $i = 1, \dots, N$, $h = 1, \dots, H$, to CPD-PO; $\pi_{k+1} \leftarrow$ its next policy. primal–dual update
12: **end for**
13: **return** $(\pi_k)_{k=1}^K$ and their uniform mixture $\bar{\pi}_{\mathcal{H}}$.

Flat-PD. Flat-PD (Algorithm 3) is CPD-PO with the same inputs (A.5), run on $\mathcal{M}$ of flat communication, whose actions are the flat step policies of Definition A.1. It differs from Algorithm 2 only in the communication round, which is the two-stage round of Section 3.2: receiver i broadcasts a b_q-bit context or stays silent (line 6); every teammate $j \neq i$ that receives the context replies with a b_r-bit message or stays silent (line 7); and receiver i acts on all $N-1$ reply slots (line 8). No sender is selected and a reply is not restricted to an observed entry. The step costs follow the flat line of Eq. (A.1), and the rescaling uses $\alpha_{\mathcal{F}}$.

Output policy. For the episode-k policy $\pi_k = \pi_{\mathrm{alg},k}$ of Section 6, $\bar{V}_k^{\mathrm{alg}} = \bar{V}^{\pi_k}$ and $\bar{C}_k^{\mathrm{alg},i} = \bar{C}^{\pi_k,i}$ are its expected return and agent-i communication cost. After K episodes the learner outputs $\bar{\pi}_{\mathrm{alg}}$, which draws an index k uniformly from $\{1, \dots, K\}$ with the shared randomness at the start of an episode and then follows π_k. Randomizing between policies at the episode's start stays within Π_{alg} (proof of Proposition A.5), and by linearity of expectation $\bar{V}^{\bar{\pi}_{\mathrm{alg}}} = K^{-1} \sum_k \bar{V}_k^{\mathrm{alg}}$ and $\bar{C}^{\bar{\pi}_{\mathrm{alg}},i} = K^{-1} \sum_k \bar{C}_k^{\mathrm{alg},i}$.

Algorithm 3 FLAT-PD. At each step h, lines 6–8 run for every receiver i in parallel.

Input: Step-policy sets $\overline{\mathcal{A}}_1, \dots, \overline{\mathcal{A}}_H$ of flat communication; $\bar{S}, \bar{A}, \alpha_{\mathcal{F}}, K, \delta$; the inputs of Eq. (A.5). The initial distribution, transitions, and rewards are unknown.

1: Initialize CPD-PO (Stradi et al., 2025, Algorithm 2) on $\mathcal{M}$ of flat communication (Definition A.2) with the inputs of Eq. (A.5); $\pi_1 \leftarrow$ its initial policy.
2: **for** episode $k = 1, \dots, K$ **do**
3: Reset the environment and the local histories; $\bar{s}_{k,1} \leftarrow \varnothing$.
4: **for** step $h = 1, \dots, H$ **do**
5: $\Gamma_{k,h} \sim \pi_k(\cdot \mid \bar{s}_{k,h})$ by shared randomness. step policy
6: $q_h^i \leftarrow \Gamma_{k,h}^{i,\mathrm{q}}(I_h^i) \in \{0,1\}^{b_q} \cup \{\perp\}$; if $q_h^i \neq \perp$, deliver it to every $j \neq i$; else set $m_h^i \leftarrow (\perp)_{j \neq i}$ and go to line 8. broadcast
7: Every $j \neq i$: $m_{h,j\to i} \leftarrow \Gamma_{k,h}^{j\to i,\mathrm{r}}(I_h^j, q_h^i) \in \{0,1\}^{b_r} \cup \{\perp\}$; $m_h^i \leftarrow (m_{h,j\to i})_{j \neq i}$. replies
8: $a_h^i \leftarrow \Gamma_{k,h}^{i,\mathrm{a}}(I_h^i, m_h^i)$. act
9: Execute $\mathbf{a}_h$; record the reward $r_{k,h}$ and the step costs $c_{k,h}^i$ of Eq. (A.1); advance the local histories; $\bar{s}_{k,h+1} \leftarrow (\bar{s}_{k,h}, \Gamma_{k,h})$.
10: **end for**
11: Pass the visited pairs $(\bar{s}_{k,h}, \Gamma_{k,h})$ with rewards $r_{k,h}$ and rescaled costs $N c_{k,h}^i / \alpha_{\mathcal{F}}$, $i = 1, \dots, N$, $h = 1, \dots, H$, to CPD-PO; $\pi_{k+1} \leftarrow$ its next policy. primal–dual update
12: **end for**
13: **return** $(\pi_k)_{k=1}^K$ and their uniform mixture $\bar{\pi}_{\mathcal{F}}$.

A.4.3 THE ASSUMPTIONS OF STRADI ET AL. (2025)

Fix $\mathrm{alg} \in \{\mathcal{H}, \mathcal{F}\}$. CPD-PO is designed for the loop-free constrained MDPs of Stradi et al. (2025, Section 3.1): the states are partitioned into layers that every episode traverses in order, the first and the last layer each hold a single state, all states share one action set, and in each episode the reward and each cost are drawn as vectors over state–action pairs with entries in $[0, 1]$. Their bounds depend on the number of layers in which actions are taken, the numbers of states and actions, the number of constraints, the Slater margin, and the number of episodes. The step-policy MDP $\mathcal{M}$ of Definition A.2 takes this form after two changes that affect no policy and no value. First, a terminal state, reached from every prefix in $\overline{\mathcal{S}}_H$, forms the last layer; the first layer is $\overline{\mathcal{S}}_1 = \{\varnothing\}$. Second, since $|\overline{\mathcal{A}}_h| \leq \bar{A}$, each set $\overline{\mathcal{A}}_h$ is written as a list of $\bar{A}$ step policies in which entries may repeat, so that every state has $\bar{A}$ actions; the distributions over this list induce exactly the distributions over $\overline{\mathcal{A}}_h$, and repeated entries have the same rewards and costs. The reward vector of episode k is the episode's reward table, and the cost vector of constraint i is agent i's rescaled communication-cost table (end of Appendix A.2.3). The constraint thresholds all equal H. The positive part of a real number is written $[\cdot]_+$.

Lemma A.9 (The assumptions of Stradi et al. (2025) hold)**.** For each $\mathrm{alg} \in \{\mathcal{H}, \mathcal{F}\}$: (i) with these two changes, $\mathcal{M}$ is a loop-free constrained MDP with H layers, at most $\bar{S} + 1$ states, $\bar{A}$ actions, and N constraints whose thresholds H lie in the allowed range $[0, H]$; (ii) its reward and cost vectors are drawn independently across episodes from fixed distributions with entries in $[0, 1]$ and means $\bar{r}$ and $\bar{c}^1, \dots, \bar{c}^N$, independently of the trajectory, and the learner has bandit feedback; (iii) Slater's condition (Stradi et al., 2025, Assumption 1) holds with margin H; (iv) the policies of $\mathcal{M}$ represent exactly the policies of Π_{alg}, with the same returns and per-agent costs and with rescaled costs $\bar{V}^{\pi}(\bar{c}^i) = N\bar{C}^{\pi,i}/\alpha_{\mathrm{alg}}$, and the constrained optimum of $\mathcal{M}$ is $\bar{V}^{\star}_{\mathrm{alg}}$.

Proof. (i) The layers are $\overline{\mathcal{S}}_1, \dots, \overline{\mathcal{S}}_H$ and the terminal state, so there are at most $|\overline{\mathcal{S}}| + 1 \leq \bar{S} + 1$ states; the rest holds by construction. (ii) By the random-tape construction at the end of Appendix A.2.3, the episode tables are independent draws from one fixed joint distribution with entries in $[0, 1]$ and means $\bar{r}, \bar{c}^1, \dots, \bar{c}^N$, and prefix selection uses only the shared randomness, which is independent of the current tape; the learner receives only the visited entries, after the episode. The model of Stradi et al. (2025, Section 3.1) samples the reward vector and each cost vector as a whole and requires no independence between entries, and their concentration bounds (Stradi et al., 2025, Lemmas 1 and 2) apply Hoeffding's inequality to each pair and each constraint separately and combine them by union bounds. These bounds use only that each entry is drawn independently across episodes and independently of the visits, which holds here; reward and cost entries of the same episode may be correlated. (iii) Let $\Gamma_h^{\circ} \in \overline{\mathcal{A}}_h$ be the step policy whose request rules all return

silence. A reply is sent only in response to a delivered request (Section 3.2), so a step under Γ_h° delivers no message and every agent's step cost is zero. The policy that plays Γ_h° at every prefix therefore has $\bar{V}^\pi(\bar{c}^i) = 0 < H$ for every i, so Slater's condition holds, and since $\bar{V}^\pi(\bar{c}^i) \geq 0$ for every π and i, the margin $\max_\pi \min_i\big(H - \bar{V}^\pi(\bar{c}^i)\big)$ equals H. (iv) The distributions over the list of $\bar{A}$ actions at a prefix induce exactly the distributions over $\overline{\mathcal{A}}_h$, with the same values; by Proposition A.5, these are exactly the policies of Π_{alg}, with the same trajectory law, return, and per-agent communication costs. By Definition A.3, $\bar{V}^\pi = \bar{V}^\pi(\bar{r})$ and $\bar{C}^{\pi,i} = (\alpha_{\mathrm{alg}}/N)\bar{V}^\pi(\bar{c}^i)$, so agent i's constraint in Eq. (A.3) reads $\bar{V}^\pi(\bar{c}^i) \leq H$, and the constrained optimum is the optimal value $\bar{V}^\star_{\mathrm{alg}}$ of Eq. (A.3). □

Theorems 1 and 2 of Stradi et al. (2025) state their bounds on the strong regret and the strong violation in $\widetilde{O}$ form; the strong regret is $\sum_{k=1}^K \big[\bar{V}^\star_{\mathrm{alg}} - \bar{V}^{\pi_k}\big]_+$, the sum of the positive per-episode shortfalls of the return, and the strong violation is the maximum over the constraints of the summed positive violations. Let Reg_K and Vio_K denote bounds of the stated orders on the strong regret and the strong violation that these theorems guarantee, which are nondecreasing in the numbers of states and actions, evaluated at H layers, $\bar{S}+1$ states, $\bar{A}$ actions, N constraints, Slater margin H, K episodes, and confidence δ' of Eq. (A.5), and define

$$\mathrm{Err}_K = \max\big\{H\,\mathrm{Reg}_K,\ \mathrm{Vio}_K\big\}. \tag{A.6}$$

With these values, Theorems 1 and 2 of Stradi et al. (2025) give, using $\bar{S}+1 \leq 2\bar{S}$,

$$\mathrm{Reg}_K = \widetilde{O}\big(H^3 N(\bar{S}+1)\sqrt{\bar{A}K}/H + H^5 N/H\big) = \widetilde{O}\big(NH^2\bar{S}\sqrt{\bar{A}K} + NH^4\big),$$
$$\mathrm{Vio}_K = \widetilde{O}\big(H^3 N(\bar{S}+1)\sqrt{\bar{A}K} + H^5 N\big) = \widetilde{O}\big(NH^3\bar{S}\sqrt{\bar{A}K} + NH^5\big).$$

Hence $\mathrm{Err}_K = \widetilde{O}\big(NH^3\bar{S}\sqrt{\bar{A}K} + NH^5\big)$; it depends only on $N, H, \bar{S}, \bar{A}, K, \delta$ and is the same for both protocols.

Lemma A.10 (HICOMM-PD and FLAT-PD)**.** Fix $\mathrm{alg} \in \{\mathcal{H}, \mathcal{F}\}$, and suppose Assumption A.7, $K \geq 1$, and $\delta \in (0,1)$. With probability at least $1 - \delta/2$, HICOMM-PD (for $\mathrm{alg} = \mathcal{H}$) or FLAT-PD (for $\mathrm{alg} = \mathcal{F}$) satisfies, for every agent i,

$$\sum_{k=1}^K \big[\bar{V}^\star_{\mathrm{alg}} - \bar{V}^{\mathrm{alg}}_k\big]_+ \leq \frac{\mathrm{Err}_K}{H}, \qquad \sum_{k=1}^K \Big[\bar{C}^{\mathrm{alg},i}_k - \frac{B_{\mathrm{alg}}}{N}\Big]_+ \leq \frac{\alpha_{\mathrm{alg}}}{N}\,\mathrm{Err}_K.$$

Proof. By Lemma A.9, $\mathcal{M}$ satisfies the assumptions of Theorems 1 and 2 of Stradi et al. (2025), and the learner is CPD-PO run on $\mathcal{M}$ with the true Slater margin H and confidence δ'. Theorem 1 of Stradi et al. (2025) gives, with probability at least $1 - c_0\delta'$, the strong-regret bound $\sum_k [\bar{V}^\star_{\mathrm{alg}} - \bar{V}^{\pi_k}(\bar{r})]_+ \leq \mathrm{Reg}_K$, since $\bar{V}^\star_{\mathrm{alg}}$ is the constrained optimum of $\mathcal{M}$ by Lemma A.9(iv), and Theorem 2 gives, with probability at least $1 - c_0\delta'$, the strong-violation bound $\max_i \sum_k [\bar{V}^{\pi_k}(\bar{c}^i) - H]_+ \leq \mathrm{Vio}_K$, so the bound holds for every i. By a union bound, both hold with probability at least $1 - 2c_0\delta' = 1 - \delta/2$. By Lemma A.9(iv), $\bar{V}^{\pi_k}(\bar{r}) = \bar{V}^{\mathrm{alg}}_k$ and $\bar{C}^{\mathrm{alg},i}_k - B_{\mathrm{alg}}/N = (\alpha_{\mathrm{alg}}/N)\big(\bar{V}^{\pi_k}(\bar{c}^i) - H\big)$. The two bounds therefore read $\sum_k [\bar{V}^\star_{\mathrm{alg}} - \bar{V}^{\mathrm{alg}}_k]_+ \leq \mathrm{Reg}_K \leq \mathrm{Err}_K/H$ and $\sum_k [\bar{C}^{\mathrm{alg},i}_k - B_{\mathrm{alg}}/N]_+ \leq (\alpha_{\mathrm{alg}}/N)\mathrm{Vio}_K \leq (\alpha_{\mathrm{alg}}/N)\mathrm{Err}_K$, by Eq. (A.6). □

A.4.4 PROOF OF THE THEOREM

Proof of Theorem 6.1. Apply Lemma A.10 separately to the two protocols with the common K and δ and the common size bounds $\bar{S}, \bar{A}$. Each protocol's event fails with probability at most $\delta/2$; a union bound gives simultaneous probability at least $1 - \delta$, without requiring independent runs. The quantity Err_K of Eq. (A.6) depends only on $N, H, \bar{S}, \bar{A}, K, \delta$ and is the same for both protocols. On this event, since $x \leq [x]_+$ for every real x, for every agent i,

$$\sum_{k=1}^K \bar{V}^{\mathrm{alg}}_k \geq K\,\bar{V}^\star_{\mathrm{alg}} - \sum_{k=1}^K \big[\bar{V}^\star_{\mathrm{alg}} - \bar{V}^{\mathrm{alg}}_k\big]_+ \geq K\,\bar{V}^\star_{\mathrm{alg}} - \frac{\mathrm{Err}_K}{H},$$

$$\sum_{k=1}^K \bar{C}^{\mathrm{alg},i}_k \leq \frac{KB_{\mathrm{alg}}}{N} + \sum_{k=1}^K \Big[\bar{C}^{\mathrm{alg},i}_k - \frac{B_{\mathrm{alg}}}{N}\Big]_+ \leq \frac{\alpha_{\mathrm{alg}}}{N}\big(KH + \mathrm{Err}_K\big),$$

using $B_{\mathrm{alg}} = \alpha_{\mathrm{alg}} H$. The flat case of the first bound is the flat reward bound of Eq. (6.2), and the second is, for each protocol and agent, its communication-cost bound of Eq. (6.2); summing it over the N agents gives the team bound $\sum_k \sum_{i=1}^{N} \bar{C}_k^{\mathrm{alg},i} \leq \alpha_{\mathrm{alg}}(KH + \mathrm{Err}_K)$. For HICOMM, substitute $\bar{V}_{\mathcal{H}}^{\star} = \bar{V}_{\mathcal{F}}^{\star} - \Delta$ (Definition A.6), which gives the HICOMM bound of Eq. (6.2). This substitution is an identity, so it requires no additional failure probability. The two reward bounds therefore differ by exactly $K\Delta$, and the two communication-cost bounds in bits differ by the factor $B_{\mathcal{F}}/B_{\mathcal{H}} = \alpha_{\mathcal{F}}/\alpha_{\mathcal{H}} = N - 1$ of Section 4.1; these ratios compare the stated expressions and do not order the two learners' realized regret, costs, or returns. □

Reading the bounds. This paragraph adds technical details to Remark 6.2. Lemma A.10 bounds the strong regret and the strong violation of Stradi et al. (2025), which sum only the positive per-episode shortfalls of the return and excesses of the cost. Every policy of Π_{alg} satisfies all N constraints, so $\bar{V}_k^{\mathrm{alg}} \leq \bar{V}_{\mathrm{alg}}^{\star}$ in every episode, the strong regret equals the regret, the realized strong violation is zero, and $\sum_k \bar{C}_k^{\mathrm{alg},i} \leq \alpha_{\mathrm{alg}} KH/N$ holds on every sample path, which is stronger than the cost bound of Eq. (6.2). The factor N in Err_K comes from the N per-agent constraints of the MA-CMDP; a single team constraint $C^{\pi} \leq B_{\mathrm{alg}}$ removes it, but bounds only the team cost. The rate of Err_K is that of the general theorems, which do not exploit that the prefix transitions of $\mathcal{M}$ are deterministic and known. Divided by K, Eq. (6.2) also bounds the return and the per-agent communication costs of the output policy $\bar{\pi}_{\mathrm{alg}}$ of Appendix A.4.2.

B Experimental Details and Additional Results

This appendix gives the implementation and training details that Section 4 omits, the experimental setup behind Section 5, and the results that the main text summarizes. Appendix B.1 states how masks and replies are implemented on the benchmarks, the training objective, the selection function sel, the written observation in training, the auxiliary loss and a bias of its targets, and the hyperparameters. Appendix B.2 states the evaluation protocol, the reported statistics, and the baselines. Appendices B.3–B.5 give each benchmark's configuration and full results: `Impact` counts and two further references on CC4, QMIX (Rashid et al., 2020) results and win rates on SMACv2, and win rates on GRF. Appendices B.6 and B.7 give the setup of the CC4 ablations and of two further tests, and Appendix B.8 lists compute, software, and availability.

B.1 Implementation and Training Details

Masks and replies. HiComm consumes only the agents' own masks: the sender restricts its group and entity choices to the groups and entities that it observes, and the receiver writes a reply only into an entry that it does not observe (Eqs. (4.1) and (B.3)). Appendices B.3–B.5 therefore specify only when an entry is observed; every other entry of the shared observation layout (Appendix B.2) is empty until a reply fills it. On SMACv2 and GRF, the returned entry can lie in any group in which the sender observes an entity, including the opponent group, not only in the sender's own ally group. In both, the ally rows of an observation exclude the observing agent, so the same row can hold different units for different agents. Entries are therefore matched by unit: a reply is written into the receiver's row of the unit that the sender's row holds. An agent has no row for its own unit, so a sender cannot return its own unit, and for writing and for the targets of Eq. (B.4) the receiver's own unit counts as an entry of M_h^i, so a reply about it is not written. CC4 host features are accumulated knowledge, while some SMACv2 and GRF features, such as relative distance and position, are observer-relative; a filled entry carries the sender's view of the entity without coordinate transformation, and the receiver's action policy learns from such entries as they are.

Training objective. The PPO runs use IPPO or MAPPO actor–critic updates (Schulman et al., 2017; Schroeder de Witt et al., 2020; Yu et al., 2022). Each update $t = 1, \dots, T$ of Algorithm 1 minimizes, over all networks and on a buffer $\mathcal{D}$ of rollouts of at most H steps,

$$\mathcal{L} = \mathcal{L}_{\mathrm{PPO}} + c_{\mathrm{V}}\,\mathcal{L}_{\mathrm{V}} + c_{\mathrm{aux}}\,\mathcal{L}_{\mathrm{aux}}, \qquad \mathcal{L}_{\mathrm{PPO}} = \mathcal{L}_{\mathrm{clip}} - c_{\mathrm{Ent}}\,\mathrm{Ent}_{\mathrm{pol}}. \tag{B.1}$$

The PPO loss $\mathcal{L}_{\mathrm{PPO}}$ of the actor (Section 4.2) consists of the clipped surrogate $\mathcal{L}_{\mathrm{clip}}$ for the action policies $\pi_{\mathrm{pol}} = (\pi_{\mathrm{pol}}^i)_{i=1}^N$, with the clipping width of Table 8, and an entropy bonus with coefficient c_{Ent} on $\mathrm{Ent}_{\mathrm{pol}}$, the mean entropy of the agents' action distributions over the transitions of $\mathcal{D}$. On SMACv2 and GRF, all agents share the parameters of π_{pol}. The shared policy also reads the written observations from earlier steps and is trained by backpropagation through full episodes. The critic reads the agent's local observation under IPPO and the global state under MAPPO and is trained by the value loss $\mathcal{L}_{\mathrm{V}}$ with coefficient c_{V}; its input is detached from the communication modules, so $\mathcal{L}_{\mathrm{V}}$ does not train them. The auxiliary term $\mathcal{L}_{\mathrm{aux}}$, with weight c_{aux}, is protocol-specific and trains the communication modules. The PPO loss depends on the communication modules only through the messages that reach the actor, so each protocol must supply a gradient path from the actor input back to its communication rules. All methods use this objective with the same backbone, hyperparameters, and observation layout; they differ in $\mathcal{L}_{\mathrm{aux}}$ and in the protocol, and HiComm uses $\mathcal{L}_{\mathrm{aux}} = \mathcal{L}_{\mathrm{ig}}$ of Eq. (B.4) below.

On SMACv2 and GRF, the update differs from this description in three ways. The buffer $\mathcal{D}$ is not emptied between updates: it holds the 32 most recent episodes, training starts once it is full, and each new episode then triggers one update on all of $\mathcal{D}$, so an episode enters up to 32 updates. Each update makes at most four passes over $\mathcal{D}$ and stops early once the approximate Kullback–Leibler (KL) divergence between π_{pol} and the action policies before the update exceeds a target. The entropy coefficient adapts to the mean entropy: c_{Ent} is replaced by $c_{\mathrm{Ent}} \max\{1,\, \mathrm{Ent}_{\min} / \max(\mathrm{Ent}_{\mathrm{pol}}, 10^{-6})\}$ with an entropy floor $\mathrm{Ent}_{\min}$, and while $\mathrm{Ent}_{\mathrm{pol}} < \mathrm{Ent}_{\min}/2$ the clipped surrogate is dropped from $\mathcal{L}_{\mathrm{PPO}}$.

The QMIX runs on SMACv2 replace $\mathcal{L}_{\mathrm{PPO}} + c_{\mathrm{V}}\,\mathcal{L}_{\mathrm{V}}$ in Eq. (B.1) by the temporal-difference loss of QMIX (Rashid et al., 2020) and minimize the sum in one backward pass; the selections are trained

by the gradient of the temporal-difference loss, passed through the straight-through estimator, and by the auxiliary loss.

Selection function. The selection function $\mathrm{sel}(q, \psi, \mathcal{C})$ of Section 4.2 and Algorithm 1 is implemented by the straight-through Gumbel-Softmax estimator (Jang et al., 2017). For a query q, a finite nonempty set $\mathcal{C}$ of candidates, and keys $\psi(c)$, let $\mu = (\mu_c)_{c\in\mathcal{C}}$ with $\mu_c = \langle q, \psi(c)\rangle$ be the vector of scores. For independent draws $\nu_c \sim \mathrm{Gumbel}(0, 1)$, $c \in \mathcal{C}$, define

$$\bar{\sigma}(\mu) = \mathrm{softmax}(\mu + \nu), \qquad \sigma(\mu) = \mathrm{onehot}\Big(\arg\max_{c\in\mathcal{C}} (\mu_c + \nu_c)\Big) + \bar{\sigma}(\mu) - \mathrm{sg}[\bar{\sigma}(\mu)], \quad \text{(B.2)}$$

where sg is the stop-gradient; $\bar{\sigma}$ is the Gumbel-Softmax relaxation at temperature 1, the value used in all runs (Table 8). Then $\mathrm{sel}(q, \psi, \mathcal{C})$ is the candidate $c^\star = \mathrm{argmax}_{c\in\mathcal{C}}(\mu_c + \nu_c)$, represented by the one-hot vector $\sigma(\mu)$, and $\mathrm{sel}(q, \psi, \varnothing) = \perp$. By the Gumbel-max property, $c^\star$ has distribution $\mathrm{softmax}(\mu)$. At evaluation, $\mathrm{sel}(q, \psi, \mathcal{C})$ is instead the candidate $\mathrm{argmax}_{c\in\mathcal{C}}\, \mu_c$ with the largest score. In the forward pass $\sigma(\mu)$ equals the one-hot vector of $c^\star$, and in the backward pass it has the Jacobian of $\bar{\sigma}(\mu)$. Algorithm 1 (lines 3–5) calls sel with the query q_h^i and the keys of Section 4.2.

Written observation in training. In training, the written observation is computed from the one-hot outputs $\sigma(\mu)$ of the selections (Eq. (B.2)):

$$\widetilde{o}_h^i[g, e] = o_h^i[g, e] + \mathbf{1}\{(g, e) \notin M_h^i\}\, \beta_h^i\, \sigma_h^i[g, e]\, o_h^{j_h^i}[g, e]. \quad \text{(B.3)}$$

Here $\sigma_h^i[g, e]$ is zero for $g \neq g_h^i$. For $g = g_h^i$, it is the component at g of the output of the sender's group selection times the component at e of the output of its entity selection, and it is zero for the entities that the sender does not observe. The weight β_h^i is the component at $\widehat{g}_h^i$ of the output of the receiver's group selection times the component at $\widehat{e}_h^i$ of the output of its teammate selection. The second term is zero if the receiver is silent, and $\sigma_h^i = 0$ if the reply is the null reply. In the forward pass, the selected components are one and all others are zero, and Eq. (B.3) gives Eq. (4.1). In the backward pass, each candidate entity of the selected group is weighted by its own component and written at its own index. The gradient with respect to $\sigma_h^i[g_h^i, e]$ is the first-order change of the PPO loss when the sender's features of entry (g_h^i, e) are written; it is zero for the entries that the receiver observes. Through the group component of σ_h^i and through β_h^i, the gradient reaches the scores of the sender's group selection and of the receiver's selections when the returned entry is one that the receiver does not observe. The rollout stores the selected candidates $c^\star$ and the draws ν. The forward pass of the PPO update uses $c^\star$, and its backward pass evaluates $\bar{\sigma}$ at the current scores with the stored ν. The update uses the observations of all agents. The probability ratio of PPO contains only π_{pol}, and the selections are trained by the task gradient and the auxiliary loss.

Auxiliary loss. The auxiliary loss steers each selection toward the candidates that can return features of entries that the receiver does not observe. For a receiver-sender pair (i, j), the elementwise positive part $\big[o_h^j[g, e]\big]_+$ of the sender's features, with $[x]_+ = \max\{x, 0\}$, is taken over the entries $(g, e) \in M_h^j \setminus M_h^i$, whose replies Eq. (4.1) writes, summed over the entries that a candidate covers and over their feature dimensions, and softmax-normalized along each selection axis to form the soft targets $p_h^{\mathrm{grp},i}$ over the groups of $\mathcal{G}$ among which the selected sender chooses the group of the returned entry, $\widehat{p}_h^{\mathrm{snd},i}$ over the teammates $j \neq i$ of receiver i, and $p_h^{\mathrm{ent},i}$ over the set $\mathcal{E}_{g_h^i,h}^{j_h^i}$ of entities of the selected group that the selected sender observes. Each target is compared with the probabilities $\mathrm{softmax}(\mu)$ from which sel samples, on the same candidates (Eq. (B.2)). For the sender, these are the probabilities $\theta_h^{\mathrm{grp},i}$ and $\theta_h^{\mathrm{ent},i}$ of its group and entity selections. For the receiver, $\widehat{\theta}_h^{\mathrm{snd},i}(j) = \widehat{\theta}_h^{\mathrm{grp},i}(\widehat{g})\, \widehat{\theta}_h^{\mathrm{ent},i}(j \mid \widehat{g})$, for the group $\widehat{g}$ that contains j, is the probability that i asks teammate j, given that i requests: $\widehat{\theta}_h^{\mathrm{grp},i}$ is the softmax of the group scores over $\widehat{\mathcal{G}}^i$, and $\widehat{\theta}_h^{\mathrm{ent},i}(\cdot \mid \widehat{g})$ is the softmax of the teammate scores $\langle q_h^i, \widehat{f}_{\mathrm{e}}(\widehat{g}, \widehat{e}; u_h^i)\rangle$ over $\widehat{e} \in \widehat{\mathcal{E}}_{\widehat{g}} \setminus \{i\}$. Silence has no target; the task gradient alone trains its key $\widehat{\kappa}(\perp)$. With $\mathrm{CE}(p, \theta) = -\sum_c p_c \log \theta_c$ and the entropy $\mathrm{Ent}[p] = \mathrm{CE}(p, p)$ of a distribution p, the auxiliary loss is

$$\begin{aligned}\mathcal{L}_{\mathrm{ig}} = \mathbb{E}\Big[&\sum_{\mathrm{ax}\in\{\mathrm{grp},\mathrm{ent}\}} w_{\mathrm{ax}}\, \mathrm{CE}\big(p_h^{\mathrm{ax},i}, \theta_h^{\mathrm{ax},i}\big) + \widehat{w}_{\mathrm{snd}}\, \mathrm{CE}\big(\widehat{p}_h^{\mathrm{snd},i}, \widehat{\theta}_h^{\mathrm{snd},i}\big)\Big] \\ &- \widehat{w}_{\mathrm{spread}}\, \mathrm{Ent}\big[\langle \widehat{\theta}_h^{\mathrm{ent},i}(\cdot \mid \widehat{g}_h^i)\rangle_{\mathcal{D}}\big],\end{aligned} \quad \text{(B.4)}$$

with the expectation over the transitions of the rollout buffer $\mathcal{D}$ and the agents, the weights $w_{\mathrm{grp}}, w_{\mathrm{ent}}$ of the sender's selections, and the weight $\widehat{w}_{\mathrm{snd}}$ of the receiver's selection; the terms of the sender's selections are omitted when no entry is returned. The last term is an entropy bonus with weight $\widehat{w}_{\mathrm{spread}}$ on the average $\langle \widehat{\theta}_h^{\mathrm{ent},i}(\cdot \mid \widehat{g}_h^i)\rangle_{\mathcal{D}}$ of the receiver's teammate probabilities over the requests in $\mathcal{D}$ (the steps h and receivers i with $\widehat{g}_h^i \neq \perp$), each extended by zero to all agents (the term is zero if $\mathcal{D}$ holds no request); it spreads requests across the teammates in a group.

Target of the auxiliary loss. Each entry of $M_h^j \setminus M_h^i$ contributes the positive part of the sender's features to the weight of a candidate in the soft targets of Eq. (B.4): for the non-negative host features of CC4 this counts what the receiver lacks, whereas for the signed features of SMACv2 and GRF (relative positions and directions) negative coordinates contribute nothing, so the target favors entities whose features are positive in the sender's frame. This bias affects only the auxiliary loss $\mathcal{L}_{\mathrm{ig}}$; a target that counts the entries of $M_h^j \setminus M_h^i$ instead of summing their features would avoid it.

Configuration. Table 8 lists the default network sizes and training settings of the PPO runs per benchmark, and Table 9 those of the QMIX runs. On CC4, f_{enc} is an MLP of the flattened observation. On SMACv2 and GRF, it first embeds each row of the observation segments (Appendix B.2) as a token of width 8 (rows of different segments have different widths), applies the MLP of Table 8 to each token, and a gap encoder maps the tokens and the observability mask to the 32-dimensional u_h^i. The keys $\widehat{\kappa}$ comprise one key per group that can be a receiver's candidate and one for $\perp$ (CC4: the six zones; SMACv2: the three ally unit types; GRF: the midfield and forward role groups); receiver i scores only $\perp$ and the groups in $\widehat{\mathcal{G}}^i$ (Algorithm 1, line 3), which on CC4 are the five zones of its teammates. The keys κ comprise one key per group of $\mathcal{G}$ (CC4: the six zones and the internet; SMACv2: the three ally unit types and the opponent group; GRF: the midfield group (DM, CM, LM, RM, AM: defensive, central, left, right, and attacking midfielders), the forward group (CF: center forward), and the opponents), and every key has the query's dimension d_q. The teammate key $\widehat{f}_{\mathrm{e}}(\widehat{g}, \widehat{e}; u_h^i)$ reads u_h^i and learned 8-dimensional embeddings of $\widehat{g}$ and $\widehat{e}$. The IPPO critic reads the agent's local observation, on SMACv2 without the previous-action and identity segments; the MAPPO critic reads the global state, which on GRF is the concatenation of the three agents' observations.

B.2 Evaluation Protocol and Baselines

Evaluation. Every reported configuration is trained with three random seeds; within each CC4 run, four rollout workers use environment seeds derived from the training seed. Per seed, every entry except the CC4 `Impact` counts averages evaluation episodes of the trained policy, run greedily at the end of training (the action policy takes its most probable action, and each selection its highest-scoring candidate) against the same fixed opponent as in training; the `Impact` counts are sums over these episodes (Appendix B.3). Each seed is evaluated for 100 episodes on CC4 and for 32 episodes on SMACv2 and GRF. Each benchmark's opposing side is frozen or scripted throughout training and evaluation (CC4 plays against a fixed pretrained blue-defender checkpoint, SMACv2 against the built-in heuristic opponent at difficulty 7, and GRF against the built-in scripted opponent).

Reported statistics. Table 2 in the main text and Tables 11, 12, and 13 report means and standard deviations across seeds; Table 10 reports them for every entry it lists, including the `Impact` counts; the ablation tables (Tables 3–5) report across-seed means only. The win rates of Tables 12 and 13 and their standard deviations are rounded to the nearest percentage point. Communication cost is measured from the messages exchanged in the evaluation episodes, accumulated over each complete episode and averaged over receivers and evaluation episodes. It is the empirical counterpart of the per-agent cost $C^{\pi,i}$ of Section 3.3, averaged over agents, but it is not computed from the message widths of that section. On CC4, the FullObs entry of Table 2 corresponds to the 720-dimensional host matrix of each of the five teammates at 4 bytes per feature over 499 message rounds per 500-step episode, $5 \times 720 \times 4 \times 499$ bytes $= 7{,}017.2$ KiB; the two global scalars of the observation are not counted.

Observation layout and baselines. Each per-agent observation is a fixed-length flattened vector concatenating semantically distinct segments in a canonical order, exposed to HiComm as a list

Table 8: Default network sizes and training settings of the HICOMM PPO runs, per benchmark. I/M denote IPPO/MAPPO; GAE is generalized advantage estimation, and GRU is a gated recurrent unit; $a \to b \to c$ lists layer widths; a SMACv2 entry $x; y$ gives x for `protoss` and y for `terran` and `zerg`. On SMACv2 and GRF, each row of an observation segment is first embedded as a token of width 8 (*row* $\to$ *token* 8); the MLP of f_{enc} and the first layer of the actor act on these tokens, and a gap encoder gives u_h^i. The CC4 batch counts samples per agent; on SMACv2 and GRF it is the window of the 32 most recent episodes.

	CC4	SMACv2	GRF
Communication modules			
Observation encoder f_{enc}	MLP $722 \to 128 \to 32$	row $\to$ token 8; MLP $8 \to 32 \to 8$	row $\to$ token 8; MLP $8 \to 32 \to 8$
Encoding dimension $\dim u_h^i$	32	32	32
Query projection f_{q}	$32 \to 32 \to 16$	$32 \to 32 \to 8$	$32 \to 32 \to 8$
Query and key dimension d_q	16	8	8
Receiver's group keys $\widehat{\kappa}$ (count, with $\perp$)	7	4	3
Sender's group keys κ (count)	7	4	3
Teammate key network $\widehat{f_{\text{e}}}$	$48 \to 32 \to 16$	$48 \to 64 \to 8$	$48 \to 64 \to 8$
Entity key network f_{e}	$5 \to 64 \to 16$	$9 \to 32 \to 8$; $8 \to 32 \to 8$	$7 \to 32 \to 8$
Gumbel-Softmax temperature (constant)	1.0	1.0	1.0
Actor and critic			
Actor network π_{pol}^i	$722 \to 3 \times 1024$	token $8 \to 64$, GRU 64	token $8 \to 64$, GRU 64
Critic hidden widths	3×1024	2×256	2×256
Critic input dimension (I / M)	$722/4332$	$90/130$; $80/120$	$103/309$
PPO			
Optimizer	Adam	Adam	Adam
Learning rate, IPPO (actor / critic)	$5\times10^{-5}/5\times10^{-5}$	$3\times10^{-4}/10^{-3}$	$3\times10^{-4}/10^{-3}$
Learning rate, MAPPO (actor / critic)	$5\times10^{-5}/5\times10^{-5}$	$3\times10^{-4}/7\times10^{-4}$	$3\times10^{-4}/7\times10^{-4}$
Discount factor	0.99	0.99	0.99
GAE parameter (I / M)	$0.95/0.95$	$0.90/0.92$	$0.90/0.92$
Clipping width	0.2	0.2	0.2
Value coefficient c_{V}	1.0	0.5	0.5
Base entropy coefficient c_{Ent}	0.05	0.05	0.02
Episode limit H (steps)	500	200	400
Parallel environments	4	1	1
Batch per update	2000	32	32
Minibatches; epochs per update	16; 10	1; ≤ 4	1; ≤ 4
Gradient-norm clip	10	0.5	0.5
Environment steps	0.2M, $T = 100$	10M	10M

Table 9: Settings of the HICOMM QMIX runs on SMACv2.

Communication modules	As in the SMACv2 column of Table 8
Agent network	GRU 64, trained on whole episodes
Mixer	Embedding 32; hypernetwork of 2 layers of width 64
Replay buffer; batch	5,000 episodes; 32 episodes
Parallel environments	1
Exploration	ϵ-greedy; the random-action probability decreases linearly from 1.0 to 0.05 over the first 50,000 steps and then stays fixed
Optimizer	RMSprop; learning rate 10^{-3}, smoothing constant 0.99, stability constant 10^{-5}
Discount factor	0.99
Double Q-learning; target network	Yes; hard update every 100 episodes
Gradient-norm clip	10
Weights $w_{\text{grp}}, \widehat{w}_{\text{snd}}, w_{\text{ent}}$; $\widehat{w}_{\text{spread}}$; c_{aux}	1.0, 0.5, 1.0; 0.1; 0.1
Environment steps	10M

of (slots, features per slot) pairs called `obs_segs`; the per-benchmark layouts appear in Appendices B.3–B.5. The CACOM and T2MAC baselines are ports of their released implementations, and EC is our implementation; all three read the same `obs_segs` layout as HICOMM and are trained under IPPO and MAPPO with the actor/critic architectures, PPO hyperparameters, entropy floor, and gradient-clip values of the other policy-gradient methods. CACOM (Li & Zhang, 2024) is a learned member of $\mathcal{F}$: its receiver broadcasts a context vector computed from its observation, and each teammate may return a receiver-conditioned message computed by entity-token attention, which is a freely chosen b_r-bit reply to a b_q-bit context heard by all. In Table 2, EC's mean reward exceeds that of PARTIALOBS in all ten configurations, and HICOMM has higher mean reward and lower communication cost than EC in all ten.

B.3 CAGE CHALLENGE 4 (CC4)

CC4 is released as a blue-side training problem (Kiely et al., 2025); the reported runs train the red side through a thin wrapper stack that leaves the simulator unchanged: a `RedFlatWrapper` exposes the six red attackers as PettingZoo agents, and a `HybridFixedBlueWrapper` freezes the blue defense, throughout training and evaluation, at the released checkpoint of the single-policy baseline of Singh et al. (2025) (`marl-1policy`), in which each blue agent is trained by PPO as an independent learner. Episodes run 500 steps. Each red agent's action pairs one of ten primitive actions with one of the 9×16 target coordinates, filtered by a per-step validity mask so the policy never commits to a syntactically illegal action. The shared red reward is obtained from the blue rewards: at every step, every red agent receives the negative of the mean reward of the blue defenders that co-occur with red under the current phase's connectivity policy, which makes the co-occurring slice zero-sum under the red-side sign convention.

Each red agent observes a 722-dimensional vector: the $9\times16\times5 = 720$ host matrix of the nine subnets with 16 host slots per subnet and five per-host features (access level, recon level, `neighbours_discovered`, `is_reachable`, `deception_known`), followed by two global scalars (the current mission phase and the previous action's success flag). The subnet-host containment is native to the simulator. Each attacker acts in its own zone of subnets: five zones consist of one subnet each, and one zone consists of the three headquarters subnets. The groups are the six zones and the internet subnet, which belongs to no zone. Each zone contains its attacker as its only agent, and the internet contains no agent (Definition 3.1). The entities of a single-subnet group are its 16 host slots. The entities of the headquarters zone are the host slots of its three subnets, concatenated into 48 slots. Red knowledge accumulates over the episode, and this accumulative host knowledge determines the observation mask: agent j observes host e of group g exactly when the host's `recon_level` is at least 1 in j's belief state. `Impact` is the attack action that disrupts a host's active operational-technology service; it succeeds only on hosts red has compromised to root level through the earlier attack stages. Each seed is evaluated for 100 episodes, and Table 10 reports two counts per seed, each summed over these 100 episodes and all six red agents: the number of successful `Impact` executions and the total number of `Impact` attempts.

Table 10: CC4 results with `Impact` counts for FULLOBS, PARTIALOBS, EC, and HICOMM, together with two further references: a broadcast autoencoder and the release's scripted finite-state machine (FSM) red agent, which acts on its local observation without communication. CACOM and T2MAC appear only in Table 2. All entries are mean ± standard deviation over three seeds (training seeds for the learned methods). Reward is under the red-side convention; for the methods that also appear in Table 2 it repeats their entries there. Success and Total #Impact are the numbers of successful `Impact` executions and of `Impact` attempts, summed per seed over 100 evaluation episodes and all six red agents. Bold marks the highest reward and successful `Impact` count per backbone (IPPO, MAPPO) among the methods listed; underline marks the second highest; the FSM row is not marked.

Algorithm	Method	Reward	Success #Impact	Total #Impact
FSM	PARTIALOBS	178 ± 46	312 ± 21	$8{,}608 \pm 572$
IPPO	FULLOBS	$\mathbf{1{,}724 \pm 185}$	$\mathbf{720 \pm 445}$	$49{,}281 \pm 7{,}179$
IPPO	PARTIALOBS	$1{,}395 \pm 134$	246 ± 44	$44{,}282 \pm 10{,}230$
IPPO	Broadcast autoencoder	$\underline{1{,}603 \pm 172}$	169 ± 108	$52{,}041 \pm 8{,}700$
IPPO	EC	$1{,}454 \pm 299$	146 ± 91	$33{,}034 \pm 5{,}495$
IPPO	HICOMM (ours)	$1{,}572 \pm 107$	$\underline{262 \pm 90}$	$49{,}499 \pm 28{,}197$
MAPPO	FULLOBS	$\mathbf{2{,}163 \pm 129}$	$\mathbf{497 \pm 225}$	$50{,}296 \pm 34{,}905$
MAPPO	PARTIALOBS	$1{,}048 \pm 150$	232 ± 139	$30{,}287 \pm 26{,}809$
MAPPO	Broadcast autoencoder	$1{,}543 \pm 353$	165 ± 25	$54{,}026 \pm 26{,}211$
MAPPO	EC	$1{,}568 \pm 157$	217 ± 90	$63{,}691 \pm 45{,}642$
MAPPO	HICOMM (ours)	$\underline{1{,}917 \pm 158}$	$\underline{246 \pm 117}$	$78{,}216 \pm 76{,}990$

Table 10 adds two references to the main comparison: a broadcast autoencoder, which communicates learned reconstructions of the observation rather than raw features, and the release's scripted finite-state machine (FSM) red agent as a non-RL reference; the rewards of CACOM and T2MAC are reported in Table 2 only. The broadcast autoencoder's mean reward is above HICOMM's under IPPO ($1{,}603 \pm 172$ against $1{,}572 \pm 107$), by less than either standard deviation, and below it under MAPPO ($1{,}543 \pm 353$ against $1{,}917 \pm 158$). Under both backbones it attempts more `Impact` actions than PARTIALOBS ($52{,}041$ versus $44{,}282$ under IPPO; $54{,}026$ versus $30{,}287$ under MAPPO) but records fewer successful ones (169 versus 246; 165 versus 232).

HICOMM records more successful `Impact` executions than PARTIALOBS and has higher mean reward under both backbones (262 versus 246 successful `Impact` and $1{,}572$ versus $1{,}395$ reward under IPPO; 246 versus 232 and $1{,}917$ versus $1{,}048$ under MAPPO). These `Impact` differences are small relative to the across-seed standard deviations: HICOMM's margins over PARTIALOBS in successful `Impact` (16 under IPPO and 14 under MAPPO) are below both standard deviations, and each broadcast-autoencoder `Impact` difference above is below the larger of the two standard deviations compared, so the counts do not by themselves separate these methods. The scripted FSM attains reward 178 ± 46 with 312 ± 21 successful executions out of $8{,}608 \pm 572$ attempts. The broadcast autoencoder is an additional CC4 reference, not one of the three baselines of Table 2, and it differs from the *All communication* variant of Table 3: the autoencoder broadcasts a learned reconstruction to every other agent, whereas that variant keeps HICOMM's single-sender raw-feature replies and detaches all communication modules from the task gradient.

B.4 SMACv2

The scenarios are the 5v5 battles of the three races `protoss`, `terran`, and `zerg`, run with the released configuration (`n_units=5`, `n_enemies=5`, `step_mul=8`) against StarCraft II's built-in heuristic opponent at difficulty 7, with unit compositions resampled each episode from the race-specific distributions and start positions randomized on a 32×32 map. Actions and rewards are the benchmark defaults: a discrete head of unit-targeted attacks, `no-op`, `stop`, and the four cardinal moves under a per-step validity mask, and the damage-shaped shared reward of health-and-shield damage dealt and enemy units killed, plus a terminal win bonus. Alongside the policy-gradient backbones, the SMACv2 grid includes the monotonic value-factorization learner QMIX (Rashid et al., 2020). CACOM and T2MAC are ported from their SMACv1 (Samvelyan et al., 2019) implementations: both receive the `obs_segs` layout so their per-entity attention reads the same entity rows

as HiComm, and both are trained under each backbone with the shared policy-gradient settings; T2MAC's evidence encoder emits a Dirichlet concentration per teammate over the 11 actions, and CACOM keeps its released latent dimension 8, attention dimension 32, mutual-information weight 10^{-3}, and entropy regularizer 10^{-2}. Every (algorithm, method, race) configuration trains for 10M environment steps under QMIX, IPPO, and MAPPO alike.

The per-agent observation concatenates six segments in a canonical order: move availability (1 slot, 4 features), one row per enemy slot, one row per ally slot excluding self, the unit's own state, the previous action as a one-hot of length 11, and the agent identity as a one-hot of length 5; on the 5v5 scenarios the enemy and ally segments carry 5 and 4 rows. Every entity row shares one feature layout (a visibility bit, relative distance, relative position, current health, current shield on `protoss` only, and a unit-type one-hot), giving race-dependent widths of 9 features per row on `protoss` and 8 on `terran` and `zerg`; rows of absent or out-of-sight slots are zeroed. The groups are the three ally unit types of the race plus one collapsed opponent group, and the agents of each unit type form the set $\widehat{\mathcal{E}}_g$ of that group (Definition 3.1); unit types are redrawn at every reset, and each agent is given its teammates' unit types at the reset, as part of its first observation, which fixes the groups of each agent's teammates for the episode. Sight ranges are the benchmark's unit-type-specific defaults, and agent j observes the unit at slot e of group g exactly when it is alive, within j's sight range, and non-padding.

Table 11: SMACv2 5v5 results with QMIX (mean $\pm$ standard deviation over three training seeds): team reward and communication cost (KiB per receiver per episode). Bold marks the two highest mean rewards and two lowest mean communication costs per race, as in Table 2.

		`protoss`		`terran`		`zerg`	
Algorithm	Method	Reward	Cost (KiB)	Reward	Cost (KiB)	Reward	Cost (KiB)
QMIX	FullObs	$\mathbf{17.90 \pm 0.59}$	207.3 ± 4.3	$\mathbf{17.55 \pm 1.08}$	143.1 ± 33.1	$\mathbf{15.18 \pm 1.13}$	93.7 ± 2.9
QMIX	PartialObs	16.32 ± 0.70	$\mathbf{0 \pm 0}$	14.79 ± 0.51	$\mathbf{0 \pm 0}$	11.40 ± 2.74	$\mathbf{0 \pm 0}$
QMIX	CACOM	16.69 ± 1.24	26.0 ± 2.6	16.28 ± 1.26	15.1 ± 1.2	12.70 ± 2.49	6.9 ± 0.2
QMIX	T2MAC	17.11 ± 1.92	119.3 ± 7.9	15.39 ± 1.44	100.2 ± 5.2	12.54 ± 0.58	66.9 ± 4.1
QMIX	EC	16.54 ± 1.76	46.7 ± 6.2	15.69 ± 0.40	29.5 ± 8.6	12.69 ± 1.35	21.7 ± 2.0
QMIX	HiComm (ours)	$\mathbf{17.56 \pm 1.02}$	$\mathbf{8.6 \pm 0.7}$	$\mathbf{16.96 \pm 1.13}$	$\mathbf{6.8 \pm 0.6}$	$\mathbf{13.37 \pm 1.48}$	$\mathbf{4.3 \pm 0.3}$

Table 12: SMACv2 5v5 win rates (%) after 10M environment steps: mean $\pm$ standard deviation over three training seeds, in percentage points. Bold marks the highest mean win rate per backbone and race; underline marks the second highest; tied entries share the mark.

Algorithm	Method	`protoss`	`terran`	`zerg`
QMIX	FullObs	$\mathbf{84 \pm 6}$	$\mathbf{84 \pm 14}$	$\mathbf{64 \pm 7}$
QMIX	PartialObs	71 ± 5	66 ± 6	43 ± 20
QMIX	CACOM	75 ± 11	75 ± 6	48 ± 16
QMIX	T2MAC	75 ± 9	70 ± 9	46 ± 7
QMIX	EC	73 ± 10	70 ± 7	46 ± 8
QMIX	HiComm (ours)	$\underline{79 \pm 7}$	$\underline{78 \pm 3}$	$\underline{53 \pm 8}$
IPPO	FullObs	$\mathbf{77 \pm 12}$	$\mathbf{77 \pm 11}$	$\mathbf{63 \pm 9}$
IPPO	PartialObs	51 ± 13	57 ± 7	35 ± 7
IPPO	CACOM	68 ± 5	61 ± 4	44 ± 3
IPPO	T2MAC	58 ± 10	58 ± 7	44 ± 9
IPPO	EC	55 ± 15	58 ± 12	43 ± 10
IPPO	HiComm (ours)	$\underline{69 \pm 11}$	$\underline{74 \pm 9}$	$\underline{51 \pm 10}$
MAPPO	FullObs	$\mathbf{79 \pm 5}$	$\mathbf{76 \pm 7}$	$\mathbf{68 \pm 17}$
MAPPO	PartialObs	57 ± 8	48 ± 8	44 ± 8
MAPPO	CACOM	65 ± 10	63 ± 3	51 ± 13
MAPPO	T2MAC	66 ± 6	53 ± 3	47 ± 6
MAPPO	EC	67 ± 2	53 ± 8	46 ± 2
MAPPO	HiComm (ours)	$\underline{76 \pm 2}$	$\underline{67 \pm 10}$	$\underline{57 \pm 11}$

Following the benchmark's win criterion (Ellis et al., 2023), an evaluation episode is a win exactly when every enemy unit dies strictly before the horizon and at least one ally unit remains alive; horizon timeouts and ally wipes count as losses. Table 11 reports reward and communication cost under QMIX: HiComm records the highest reward among the four communication methods on every race, ahead of CACOM by 0.87, 0.68, and 0.67 reward points, of T2MAC by 0.45, 1.57, and 0.83, and of EC by 1.02, 1.27, and 0.68 on `protoss`, `terran`, and `zerg`; only the margins over T2MAC and EC on `terran` are larger than both methods' standard deviations. It communicates the least, at 8.6, 6.8, and 4.3 KiB per receiver per episode against CACOM's 26.0, 15.1, and 6.9, T2MAC's 119.3, 100.2, and 66.9, and EC's 46.7, 29.5, and 21.7.

Table 12 consolidates win rates across the three backbones: HiComm's margins over CACOM are 4, 3, and 5 percentage points under QMIX, 1, 13, and 7 under IPPO, and 11, 4, and 6 under MAPPO; its margins over T2MAC are 4, 8, and 7 under QMIX, 11, 16, and 7 under IPPO, and 10, 14, and 10 under MAPPO; and its margins over EC are 6, 8, and 7 under QMIX, 14, 16, and 8 under IPPO, and 9, 14, and 11 under MAPPO. HiComm's win rate is below that of FullObs and above that of PartialObs in all nine backbone–race cells. Under QMIX, only the margin over EC on `terran` exceeds both methods' standard deviations. Under IPPO, the margins over all three baselines on `terran` do; under MAPPO, the margins over all three on `protoss` and over T2MAC and EC on `terran` do.

B.5 Google Research Football (GRF)

The sole GRF scenario is `academy_counterattack_hard` (4v3): a fast break in which the controlled side fields three policy-controlled attackers and a goalkeeper handed to the engine's scripted policy following the academy protocol of Song et al. (2024), so $N = 3$, against a fully scripted right team, with episodes capped at 400 steps. The action space is the GRF default 19-action discrete head (`idle`, eight directional moves, three passes, `shot`, `sprint`, `slide`, `dribble`, and the release actions of the sticky moves) under a per-step validity mask. Training uses the `scoring,checkpoints` reward of Kurach et al. (2020), as in Song et al. (2024), averaged over the three agents. The CACOM and T2MAC ports reuse the SMACv2 port settings, with the 19-way action head as the only environment-specific change. Every (algorithm, method) configuration trains for 10M environment steps under IPPO and MAPPO.

The per-agent observation is a 103-dimensional entity-based vector of six segments: the agent's own state (29 features), the ball state (18), one row per enemy slot, one row per ally slot excluding self (the other attackers and the scripted goalkeeper), and one summary row each for the closest teammate and the closest opponent; on `academy_counterattack_hard` the enemy and ally segments carry three rows each. Every player row carries the same seven features (scaled position, direction, speed, distance to the agent, and tiredness), so one reply format serves ally and opponent rows alike. The ally groups are the midfield and forward role groups, read from the observation's role one-hot, plus one collapsed opponent group; the agents of each role group form its set $\widehat{\mathcal{E}}_g$. Agent j observes the player at slot e of group g exactly when that player lies within j's sensing radius and the slot is non-padding.

Table 13 complements the reward comparison of Table 2 with goal-based win rates, the GRF community convention, over the final greedy evaluation episodes; the win rate depends only on the scoring component of the reward and is invariant to the checkpoint shaping. Under IPPO, HiComm records 56% against 52% for EC and 50% for each of CACOM and T2MAC, above PartialObs at 48% and below FullObs at 61%; the rewards of Table 2 give the same order (FullObs 1.42, HiComm 1.35, EC 1.18, CACOM and T2MAC 1.15, PartialObs 1.05). Under MAPPO, HiComm records 68%, below FullObs at 70% and ahead of CACOM's 64%, EC's 61%, T2MAC's 60%, and PartialObs's 54%; the rewards of Table 2 again give the same order (1.64, 1.53, 1.35, 1.31, 1.26, and 1.15). The win-rate standard deviations range from 3 to 13 percentage points; each of HiComm's margins over the three baselines is smaller than the larger of the two standard deviations, whereas its MAPPO margin over PartialObs (14 points) exceeds both.

Table 13: GRF `academy_counterattack_hard` win rates (%) after 10M environment steps: mean ± standard deviation over three training seeds, in percentage points. Bold marks the highest win rate per backbone; underline marks the second highest; tied entries share the mark.

Algorithm	Method	Win rate
IPPO	FULLOBS	$\mathbf{61 \pm 13}$
IPPO	PARTIALOBS	48 ± 4
IPPO	CACOM	50 ± 3
IPPO	T2MAC	50 ± 5
IPPO	EC	52 ± 5
IPPO	HICOMM (ours)	$\underline{56 \pm 8}$
MAPPO	FULLOBS	$\mathbf{70 \pm 7}$
MAPPO	PARTIALOBS	54 ± 10
MAPPO	CACOM	64 ± 13
MAPPO	T2MAC	60 ± 9
MAPPO	EC	61 ± 8
MAPPO	HICOMM (ours)	$\underline{68 \pm 5}$

B.6 CC4 ABLATION VARIANTS

Each variant of Table 3 changes one component of the HICOMM reference and is trained under the CC4 protocol of Appendix B.3, under IPPO and MAPPO with three seeds.

Design variants. *Encode host matrix* replaces the returned feature vector with the output of a learned message encoder over the sender's full host matrix (autoencoder-style, with capacity comparable to CACOM's message head). *Encode selected entity* passes the selected entity's 5-dimensional feature vector through $\texttt{Linear}(5{\to}8) \to \texttt{ReLU} \to \texttt{Linear}(8{\to}5)$ before it is returned. *Joint group/entity selection* replaces the sender's two-step choice (a group, then a host of that group) with one straight-through Gumbel-Softmax pick over all 144 (group, host) pairs.

Detach variants. Both variants keep the auxiliary loss. *All communication* detaches the outputs of the query and of all four selections: the query receives no training signal, and the keys train only through the auxiliary terms, except the silence key $\widehat{\kappa}(\perp)$, which has no auxiliary target, and the teammate key $\widehat{f_{\mathrm{e}}}$, whose selection on CC4 has one candidate, the attacker of the chosen zone; neither trains. *Query* detaches the output of the query, which also blocks the auxiliary gradient to f_{q}, so the query stays a fixed random function of the receiver's encoding.

B.7 CC4 FURTHER TESTS

Reassigned hosts: setting. The three HICOMM IPPO policies of Table 2, trained with the simulator's group assignment, are evaluated without retraining after a fraction $\rho \in \{5\%, 10\%, 20\%\}$ of the non-padding hosts is moved to other host slots, in the same or another group. A moved host keeps its five features and its mask entry; only its index (g, e) changes. The sender's group selection counts it in the group of its new slot, which changes the candidate set $\{g : \mathcal{E}^j_{g,h} \neq \varnothing\}$ and the hosts behind each group key $\kappa(g)$, and a receiver writes a reply about the host at its new index (Eq. (4.1)). The CC4 mask is defined by red's accumulated `recon_level`, not by group, so the senders that can return a moved host are unchanged. Each seed is evaluated greedily for 100 episodes (Appendix B.2). Feature noise and training under a perturbed or learned hierarchy are not tested.

Reassigned hosts: results. At $\rho = 5\%$ and 10% the mean reward stays above those of CACOM (1,515) and EC (1,454), both evaluated with the correct assignment (Table 14, top). At $\rho = 20\%$ the margin over PARTIALOBS (106) is smaller than the across-seed standard deviations of the reference (107) and of PARTIALOBS (134). Because Table 4 reports means only, none of the decreases is shown to exceed seed variation.

Reassignment leaves the entity keys $f_{\mathrm{e}}(o^j_h[g, e])$, which read only a host's features, and the mask unchanged. It affects the group keys $\kappa(g)$, trained on the original group composition, and the mod-

Table 14: Quantities derived from the means of Tables 2, 4, and 5 (CC4, IPPO). *Change* is relative to the reference ($\rho = 0$ or $k = 1$). *Gain kept* is $(R - R_{\mathrm{P}})/(R_{\mathrm{ref}} - R_{\mathrm{P}})$ and *Gap closed* is $(R - R_{\mathrm{P}})/(R_{\mathrm{F}} - R_{\mathrm{P}})$, with $R_{\mathrm{P}} = 1{,}395$ (PARTIALOBS), $R_{\mathrm{F}} = 1{,}724$ (FULLOBS), and $R_{\mathrm{ref}} = 1{,}572$. *Added cost* is the cost increase over $k - 1$ senders.

Reassigned ρ	0%	5%	10%	20%
Reward	1,572	1,549	1,524	1,501
Change	–	−1.5%	−3.1%	−4.5%
Gain kept	100%	87%	73%	60%
Gap closed	54%	47%	39%	32%

Senders k	1	2	3	4
Reward	1,572	1,549	1,553	1,581
Change	–	−1.5%	−1.2%	+0.6%
Cost (KiB)	10.4	19.7	28.3	38.8
Cost relative to $k = 1$	1.0	1.9	2.7	3.7
Added cost (KiB)	–	9.3	8.6	10.5

ules that read a host's position: the observation encoder f_{enc} and the actor, MLPs over the flattened 722-dimensional observation (Table 8). The test does not separate these contributions.

Several senders: setting. This test relaxes the single-sender restriction of Section 4.1, which underlies the return gap Δ of Section 6. Under IPPO, each active receiver sends its query q_h^i to $k \in \{2, 3, 4\}$ distinct teammates. Each sender applies the unchanged reply rule and returns one entry of its mask or the null reply. The receiver writes every returned entry that it does not observe at its own index (Eq. (4.1)), without aggregating or weighting replies. All other components are those of the reference. The per-agent step cost can reach $k(b_q + b_r)$ and the capacity becomes $k\,\alpha_{\mathcal{H}} = k\,\alpha_{\mathcal{F}}/(N-1)$; with $N = 6$ on CC4, $k = 5$ would reach the flat capacity. Evaluation and cost accounting follow Appendix B.2.

Several senders: results. Each added sender costs 8.6–10.5 KiB, close to the 10.4 KiB of one sender (Table 14, bottom). Every k has a mean reward above CACOM (1,515) and EC (1,454) and below T2MAC (1,620); the cost reaches that of EC (27.4 KiB) at $k = 3$, exceeds that of CACOM (33.5) at $k = 4$, and stays below that of T2MAC (79.8).

Two properties of CC4 may limit the value of extra replies: the actor is feed-forward in the current written observation (Table 8), so a reply affects only the step in which it arrives; and the k senders choose independently from the same query, so their replies can describe the same host or group.

B.8 COMPUTE, SOFTWARE, AND AVAILABILITY

All experiments ran on a single node with four NVIDIA L40S GPUs. The archived configuration records pin the software of each benchmark: for CC4, PyTorch 2.2.0, NumPy 1.26.4, Gym 0.26.2, Gymnasium 0.28.1, and PettingZoo 1.24.3; for SMACv2, StarCraft II 4.10, NumPy 1.21.4, Sacred 0.8.2, PyYAML 5.4.1, and Weights & Biases 0.13.8, with PyTorch declared but unpinned; for GRF, Python 3.9.25, `gfootball` 2.10.2, PyTorch 1.13.0, NumPy 1.23.1, Gym 0.12.4, and Weights & Biases 0.12.11. The unpinned SMACv2 PyTorch version limits exact numerical reproduction. All reported averages aggregate evaluation episodes over independently trained policies, three per configuration.

All three benchmarks are publicly accessible: CC4 through the CAGE Challenge 4 repository under its MIT/CC0 terms, SMACv2 under the MIT License with a local StarCraft II installation, and GRF under the Apache License 2.0. No human-subject, personal, or newly collected dataset is used; all reported trajectories are generated in these simulators. The anonymous submission contains the manuscript only; the authors intend to release the implementation, configuration files, trained checkpoints, and retained aggregate evaluation records after review, subject to the licenses of the benchmark and baseline repositories.