Running Supercollaborative Open Problem Sessions

Jeffrey Bosboom Erik D. Demaine Martin L. Demaine Jayson Lynch

1 Overview

v1.0 (August 7, 2018)

Supercollaboration is an unusual but effective way of conducting research, where several people work together as peers to solve unsolved problems of common interest in a positive atmosphere, without worrying about authorship and setting aside ego. Our personal experience is that the supercollaboration model is extremely effective at solving many problems (having led to many publications) while having fun (building long-lasting group camaraderie), as well as introducing students to research.

To enable these outcomes, two core rules define the supercollaboration model. First, **authorship** on papers that result from supercollaboration is self-determined by each participant and generally in alphabetical order. Second, the unsolved problems and resulting discussion are **confidential** within the group, and can be shared with others only if the group agrees to it (or when the results get published). These rules provide a safe environment where participants feel that their contributions will be valued and rewarded. (See Section 5.1 for details.)

At a high level, the main tasks to making a supercollaborative open problem session are as follows. In the sections that follow, we detail each of these tasks and suggested best practices for making things go as smoothly as possible.

1. **Assemble a group** with common interests in problems to solve.
2. **Generate open problems** that strike a balance between being easy enough to solve during the session, yet challenging enough to be interesting enough to publish.

   This is probably the most difficult part, though once a session is running smoothly, all the participants can help.
3. **Encourage and moderate** during the sessions ("driving").
4. Get people to **write**: take notes of partial progress and, upon success, write papers.

Audience. You should be reading this document if you're organizing, or interested in organizing, a supercollaborative open problem session.[1] Participants in an open problem session shouldn't need to read this document, though specific subsections will be helpful for anyone who is helping with the organizational tasks.

Research fields. Most of our experience with supercollaboration is in the field of theoretical computer science (so the examples in this document are drawn from that field). With appropriate modifications, we believe that the model can be applied to other fields as well.

Coauthor. ✎ We will frequently refer to the free software *Coauthor*,[2] which we have developed specifically to support supercollaboration by making it easy for the whole group to collaboratively take notes, recording ideas and progress as they happen (see Section 5.3). Any Coauthor-specific advice will be marked with the icon ✎.

[1] We'll use the second person "you" to refer to said organizer(s).

[2] https://github.com/edemaine/coauthor/

Contents

2 Formats

We have successfully used supercollaboration in two general meeting formats: **workshops**[3] where participants meet most of the day for multiple days (typically a week), and **weekly meetings** over a longer period (typically a semester or multiple years). You probably already have one of these two models in mind: maybe you're organizing an event that people from around the world will attend (workshop format), maybe you're considering supercollaboration within your research group (weekly meeting format), or maybe you're considering supercollaboration within your semester-long class (weekly meeting format). In that case, read the subsection relevant to your setting.

2.1 Workshop Format ("the Barbados Model")

In the workshop format, participants come together to work intensely each day for a number of days. The typical length we've worked with is one week (e.g., six days plus arrival and departure days), which comes from our 20 years of experience with the Bellairs Winter Workshop on Computational Geometry held in Barbados each year for over 30 years (hence "the Barbados model"). But we have also experimented with satellite events to conferences that run for half a day or a weekend,

[3]Supercollaboration workshops are about doing collaborative work, and generally involve no talks, in contrast to one common use of the term "workshop" to mean "small conference".

with reasonable success. The main distinction of this format is that the group exists mainly for the duration of the workshop, and to discuss follow-up work especially on paper writing and completing partial results. The event may happen regularly (e.g., yearly) but, given the timescale, the membership often varies.

Preparation. Because workshops ask a substantial time and travel commitment from participants, but also speed by, it's especially important to invest preparation time so things go as smoothly as possible during the event.

Invitations. First, you'll want to invite the right people, who will work well in and enjoy a supercollaborative environment, so that they will both contribute (hopefully) and feel good about participating. (In an event involving travel, participants cannot easily self-select and stop attending.) Most often, supercollaboration workshops are by invitation only; one exception is a satellite workshop of another event where it makes more sense to open participation to anyone at the event. See Section 3.2 on membership.

Open problem list. Second, you'll want to collect a good set of open problems for participants to work on during the workshop. Designing open problems and researching past work takes a lot of time and is less amenable to collaboration, while workshop time is precious, so it's best for workshop time to focus on solving problems. The best workshops have a list of open problems circulated before or at the very beginning; earlier distribution gives people time to familiarize themselves with the problems, often while traveling to the workshop. To help gather problems, you can ask participants (e.g., in their invitation letter) for open problem suggestions. It's still best if you edit those problems and possibly filter problems for relevance and quality, as detailed in Section 4. You can either collect the problems and assemble them yourself into a LaTeX document, or enable participants to contribute directly to the problem list.

 In the first case, you can Import the assembled LaTeX document into Coauthor for problem discussion during the workshop. In the second case, you can create a Coauthor group ahead of time and ask participants to pose problems directly there (which gives you somewhat less editorial control, but is less effort on your part). Or you can use a mix of the two, seeding with a list but encourage others to add problems.

The problem list should be short enough to cover briefly (say, within an hour) of the opening session. Ideally, the problem list has substantially more problems than could be reasonably solved during the session, to allow some selection among the participants. If you run out of problems, however, you can always ask participants for more in the middle of the workshop. For example, you can announce early one day that the workshop has made too much progress and needs more problems, so please think of some for a session later in the day. It'll be easier for participants to think up relevant problems having already worked on a bunch of such problems.

Daily schedule. Working all day every day is not a sustainable level of effort for most people, due to mental fatigue, the need to check email, and a desire to socialize (which helps with group bonding). Designating break and free time helps participants treat work-session time with respect (focusing on problem solving and collaboration with the group) because they know any distractions like email can be dealt with in the next break. Occasional small breaks during a multi-hour work session are still okay. If a participant gets tired with what they're working on, checking in with other subgroups working on other problems is a useful way of regaining focus, as well as better known methods like caffeine and snacks.

For example, Barbados workshops have morning and evening work sessions of three hours each (9–12 and 7–10), with a group dinner (6–7) and free time in the afternoon (12–6). This schedule

gives the feeling of having two days in one, and leaves lots of time for daytime activities, but makes for rather long days. During the free time, some participants continue to work, and most socialize, exercise, and/or rest. At other workshops, people work 9am–5pm with breaks for coffee and lunch, more like a traditional conference. In a single day or weekend event, you could consider even more intense days. You may need to experiment to find what format works best for you, depending on the location, participants, and when they'd like to have their free time.

Pre-arranged or catered meals are faster than leaving the group to self-organize at mealtimes, and can be useful social events in their own right. On the other hand, having meals (especially lunch) in the same space as problem solving may encourage participants to continue working when they'd be better off long-term by taking a break. One strategy to prevent this is to have designated break time adjacent to provided meals.

Location and venue. Exciting/novel locations with lots of things to do might attract people to the event, but in our experience, somewhat boring locations lead to more work getting done, as participants aren't distracted or tired out by the tourism they do during breaks.

In addition to the main working room (discussed in Section 3.3), it's ideal for the venue to have one common space (or at most a small number of spaces) participants gather and socialize during the breaks. In particular, this makes it easy for participants to find and interact with each other. Communal living is also helpful (provided most participants don't spend a lot of break time in their rooms), and a powerful agent for group bonding.

Wrap-up session. Because workshops are naturally short-lived, it's important to devote the last work session to summarizing and celebrating the progress over the entire event, and planning for continued collaboration in the future. For each solved problem/result, the group should discuss possible publication venues, and extract a volunteer to be in charge of driving the result into written form. For each problem at least briefly considered, the group can discuss how fruitful it seemed, and whether participants are interested in further work on the topic.

It's also important to establish electronic communication protocols once everyone has dispersed from the workshop. It also helps to have a clear mailing list for the entire group, and for larger groups, it can help to create sub mailing lists for specific topics/problems. Mailing lists can be especially helpful for higher-level discussion (like "should we target this deadline?") or social discussion (like "check out these photos I took during the workshop!").

If the group used Coauthor to take notes during the session, participants should be encouraged to continue to use it after the workshop, and to set their email notification settings so that they'll receive notifications of future progress on problems that interest them.

2.2 Weekly Meeting Format

In the weekly meeting format, local participants (plus maybe the occasional visitor) meet periodically over several weeks. The typical format we follow is a 2-hour meeting once per week. In the early weeks, we aim to pose one or a few problems per week, to keep things exciting and fresh, though the participants should also be encouraged to continue working on previous problems if they have ideas to continue exploring. A typical meeting starts with a review of progress during and since the last work session (see Section 5.4)¡ then a brief review of the new (and old) problems that you might work on, and then (for large groups) the (self-)assembly of participants into subgroups most interested in a particular problem.

Supercollaboration in classes. We've had particular success organizing weekly supercollaboration sessions around university classes. The topic of the class naturally makes a focused topic for supercollaboration, and the students in the class naturally form a group with shared background, knowledge, and interests (the prerequisites and the class material). After learning the latest results in the field from class, it's natural to try to extend those results or otherwise push the research envelope. The teachers also often think of related open problems when reviewing and presenting the material. Students are also motivated to participate as a way to generate final projects (if the class requires them).

We've tried two styles of weekly supercollaboration sessions associated with a class. In the simplest format, you announce an optional weekly 2-hour session for research related to the class, find a minimally conflicting meeting time (see below), and suggest anyone interested show up. Typically, a lot of students come to the first session to see what it's about, and then later sessions are much smaller. In the flipped classroom format, you have students watch videos between classes, and you devote most of the class to solving problems, but of two types: unsolved problems get tackled in the supercollaboration format, while "solved problems" (which the teaching staff knows the answer to, like questions on a problem set) get tackled in smaller groups. We've typically run these flipped classes as 2.5 hours per week, with required attendance, and the requirement to work on *something*, solved problems or unsolved problems or both. Ideally, (some of) the solved problems for a given week dovetail nicely into the week's unsolved problems.

Coauthor supports threads for both unsolved and solved problems using Thread Privacy settings ("public messages only" and "private messages only" respectively). In addition, you can allow "public and private messages" to give students a venue for feedback (e.g., on video lectures) that they can choose to share with the entire class or just instructors (similar to Piazza).

While the class serves as a catalyst for group formation, the bonding experience of solving problems sometimes leads to the group wanting to continue beyond the duration of the class. For example, the group from MIT's Algorithmic Lower Bounds class (6.890) in Fall 2014 continued meeting for over three years after the class, and occasionally still meets. Whether to continue the group is a natural topic for discussing in the final "wrap-up" work session (see the end of Section 2.1).

Membership. The weekly meeting format naturally lends itself to participants self-selecting depending on their interest and as their workload permits. We've usually followed an open membership model (see Section 3.2), in particular allowing participants who are interested but not actually taking the associated class, though usually request that participants run suggestions by us before inviting someone new.

Working between work sessions. Subject to the confidentiality principle, we encourage participants to think about and work on problems outside of the session. We have informally experimented with assigning tasks to do between sessions, but in practice we've usually found that most work occurs during the sessions. This phenomenon can be attributed to participants working more effectively together during the session than alone between sessions, or to the participants' other responsibilities taking priority.

If participants are using Coauthor to take notes during work sessions, they should similarly use it to record ideas and ask questions between sessions.

It can be nice to pose any new open problems a bit before each working session, so that only a brief review is needed during the session itself, and participants have some time to internalize the problems and maybe come up with an idea. In some situations, however, we've found that it's

better to wait until (near) the session itself, because some participants end up solving the problems before the session even starts, taking the fun away from the others.

Finding a time. Periodically (e.g., every semester), you'll need to find a time that all (or at least most) participants can attend. We typically use http://whenisgood.net/ for this purpose, as its user interface enables quick "painting" of available times, and the feedback easily shows the times with the fewest conflicts. Another classic choice is https://doodle.com/.

3 Starting an Open Problem Session (one-time tasks)

3.1 Theme

An open problem session needs a unifying theme, such as a research topic or class, that all participants are interested in. The theme cannot be too broad: "computer science" or even "theoretical computer science" or "algorithms" can be too broad, as (in a sufficiently diverse group of such participants) it can be hard to identify problems that everyone cares about. (We had one such negative experience for the topic "algorithms" in the large theory group at MIT.) Positive examples include "low-dimensional computational geometry", "geometric folding algorithms", "geometric puzzles", "recreational algorithms", "self-assembly algorithms", "hardness proofs", and "fine-grained complexity". Especially positive examples have been with the topic of an advanced semester-long (or year-long) class; see Section 2.2. A broader theme can work fine for a smaller/more-focused group, e.g., a professor's research group.

3.2 Membership

Desired qualities. As with any collaboration, who is involved is of critical importance. Fundamentally, a participant must be willing to collaborate with other people, and to buy into the authorship model and research norms set about by the open problem session, so it's important to be clear about these up front. A second essential factor, also mentioned in Section 3.1, is that the participants all share a common interest in the research topic, and they have enough shared background to be able to work together on the problems. At the same time, we desire a diversity of skills, approaches, and perspectives to tackle the research with. Beyond bringing expertise in any field, we have found far more important qualities to be general creativity, comfort with communication, and not having a personal ego. It's often more important that people get along and work well together than that they are star individual performers. Diversity within the group (including research background, experience, education level, age, sex, race) can be especially helpful for escaping traditional boundaries in problem solving.

Open model. One approach is an *open* model of participation where you invite anyone from a group, for example, students in a graduate class, all the attendees of a conference, or anyone interested in the proposed theme who can attend. Such open calls rely on self-selection and a clear description of the expectations of the open problem session.

This approach runs a higher risk of having people who will not enjoy the format (or who might abuse it). It's ideal for participants to be able to try out the open problem session for one or two meetings and then be able to leave if they don't like the rules or the format, though this is really practical only in the weekly meeting format. An open problem session run alongside a class ensures that everyone there has a shared background on the topic, and potentially have that class and their

school as a shared community to mitigate adversarial behavior. We recommend removing access to shared infrastructure like Coauthor or Github for any users who have stopped attending and made no contributions, to reduce the chance of bad behavior.

Invitation model. The other main approach is an *invitation* model of participation, where you invite individuals rather than a group, and any additional invitations need to go through you. Sometimes this model is necessary because of space or logistical limitations (e.g., in a communal living environment). It also allows you to better filter for people who you think will benefit from and participate in the supercollaboration format, which is especially helpful for week-long workshops. This model requires you to be more aware of the people you're inviting and whether they will have the common interests, backgrounds, and social norms necessary to work together on research problems.

Group size. We have used supercollaboration successfully with as few as three members and as many as 60–80 members (though the latter often involves a class where students are free to work on solved or unsolved problems). In small groups, all members can work together on the same problem. As group size increases past around eight people, it becomes difficult to all work on the same problem, both because it's hard to keep everyone on the same page and because people must take turns to talk. Larger groups (even starting around five people) naturally split into subgroups working on different problems (or different approaches to or cases of the same problem). Although participants largely self-organize into groups, the driver can help with the split by facilitating finding participants with common interests; see Section 5.2. Shared notes via Coauthor (Section 5.3) and progress reports (Section 5.4) keep participants informed about the work in other subgroups, so that they remain aware of the entire group's progress and can switch to other problems as they become interested. It's also good practice for participants to mix around periodically to interact with different subgroups, though this can depend on personality.

3.3 Meeting Room

There's more to choosing a meeting room than ensuring everyone fits in it and that everyone can access (e.g., not locked at the session time). Effective collaboration requires everyone in the (sub)group to see and hear one another. Having movable tables and chairs and enough extra room to move around allows the group to self-organize, especially when splitting into subgroups. In particular, lecture halls with front-facing chairs in fixed rows are especially poor choices for an open problem session. Especially nice are rooms that have natural alcoves or corners where subgroups can assemble. On the other hand, having at least one location in the room with unobstructed vision to the rest of the room is useful for progress reports (section 5.4) or otherwise broadcasting ideas to the whole group. It is generally better for a room to be too big than too small, even if the unused space is initially intimidating.

The room should have sufficient whiteboard/blackboard space, and the boards should be located in places where they can be read while working at tables (on paper or laptops). There should also be sufficient space and lighting to photograph the boards as part of taking notes. A projector is strongly recommended as it enables displaying notes from previous sessions (e.g., during progress reports), to project the problems while they're described, to show visualizations or demos, and to allow multiple people to read a paper or figure together.

For small groups, a typical conference or meeting room with a central table, movable chairs, one or more whiteboards, and a projector is fine. For large groups, a large open room with movable

tables and multiple whiteboards and one or more projectors is preferred. Multiple nearby rooms can also be used for very large groups, especially if they are connected by a common space and doors can remain open. Subgroups must be able to separate sufficiently so they do not interfere, but remain close enough to allow people and ideas to flow between them.

You'll want to stock up with any tools or gadgets that might be helpful for the problems you're tackling. We'll often bring paper, scissors, and tape for ad-hoc modeling, relevant props/toys (especially construction toys), and of course laptops for research, coding experiments, and note taking.

4 Designing Open Problems

Perhaps the most difficult part of running an open problem session is designing suitable open problems. Not every problem that fits the theme of the open problem session is a good open problem. What makes a problem "good" varies from field to field. Based on our experience in theoretical computer science, an ideal problem in that field has the following properties:

1. **Easy to state.**

   A rough guideline is that the problem can be stated in a single sentence. This sentence might then require some terms to be defined, but ideally those definitions aren't too many or too long (given the common knowledge of the group, e.g., from past lectures in a class). Two examples are "can every polyhedron be generally unfolded?" followed by definitions of "polyhedron" and "general unfolding", and "can every polyhedron be folded from a large enough square of paper?" followed by definitions of "polyhedron" and "folding".

   On the other hand, it's great if the problem has many variations, special cases, etc. that can be explored, in which case it's good to list those too. Each case should be succinctly statable, but it's OK if there's a lot of them, assuming you only need to understand one to get started. In the examples above, you could consider special cases of polyhedra like convex polyhedra, topologically convex polyhedra, orthogonal polyhedra, doubly covered polygons, etc.; or variations on the notion of unfolding (e.g., vertex unfolding) or folding (e.g., simple folds). (Warning: many of these specific problem have already been solved, though some are open at the time of this writing; before starting to think about any problem, research what's known.)

   It's also useful for the problem to be precise. It can be open-ended in that there can be many different goals to consider, but the problem poser should take the time to think of and list (at least some of) those goals. We've had some limited success with meta-problems that are about posing the right problem, but not when the meta-problem is overly vague. It's generally easier to brainstorm generalizing a few examples of an interesting thing (e.g., games of a genre) than specializing a general concept (e.g., applying a known technique to something new).

2. **Not too easy (and interesting).**

   If the problem is too trivial, it won't feel like an accomplishment for the group to solve it. If a solution to the problem might be publishable, that's definitely above the bar. But it also includes problems whose solution are at least interesting to someone, or fun to solve, or might appear in a nonrefereed venue (e.g., arXiv.org), or might lead to further developments (e.g., you have in mind a bigger problem to solve if this one works out). We especially like to include some easier problems when a problem session first starts, to establish some

confidence and build rapport within the group. Typically, it's enough to think about the problem for a few minutes to make sure it doesn't follow from some known result in a simple way. (You don't want to spend too much time, or else you're taking the fun away from the group!)

Relatedly, it's helpful to explain to the group (if it's not obvious) briefly why the problem is interesting — because it's fun, or because of its relation to other problems, or because of applications, etc. This will help get the participants excited about working on the problem.

3. **Not too hard.**

   The problem needs to be at least plausible to make some progress within one work session, and solve or make major progress on within the lifetime of the problem solving group. This property is of course difficult to judge until after you know what the solution is, but use your best judgment.

   It's OK to have some harder problems if there are multiple problems on the list with a variety of conjectured difficulties. You can also pose a harder problem if you can also break it down into smaller pieces or special cases that might be tractable. But you definitely don't want all your problems to be famous open problems, for example.

4. You have an **idea or approach** to solving the problem.

   This property is the least necessary, but it definitely gets the ball rolling and helps verify "not too hard". For example, you might know a solution to another problem that could be adapted to this problem. Or you might have a conjectured algorithm but not know whether it's correct. Or you might have a special case, a related problem, or a decomposition into subproblems that hasn't been explored before so might make the problem easier. Your idea doesn't need to pan out; it will still help people get into the problem and come up with other ideas. It's important for the idea to be relatively succinct to state, so you don't spend much of the collaborative time e.g. teaching people about the technique you want to apply — much better if others already know the technique (e.g., from class) or it's short.

5. **Not too much related work** (and you know it).

   The group needs to quickly come up to speed with what's known about the problem. Ideally, either the group already knows the related work (e.g., from past lectures in a class) or known results can be summarized as a black box (e.g., "known to be NP-hard, but is it in NP?"). If needed, details of related work can be presented during the session, but you'll want to keep this short (and just among the participants who are interested), so as not to take much time away from problem solving. (It's easy to get carried away and say too much about past work that might not help solve the problem.)

   An important note here is that it's very helpful for the problem poser to have done the legwork to find, understand, and summarize the related work. Most importantly, you don't want to work on a problem that turns out to have already been solved. But also you don't want to spend valuable in-session time reading past papers to figure out the results or techniques — that's best done by one person and then summarized as needed.

   More broadly, you want problems that not many people have worked on before, because those are more likely to not be too hard.

None of these properties are strictly necessary, but it's useful to think about how your problem fits according to these metrics, to make sure it has a reasonable chance of being appropriate. The

main point is that not all problems are necessarily appropriate for solving in a supercollaborative session, and you'll learn which are best through experimentation. Also, research ahead of time can go a long way toward saving in-session time.

You can also help mitigate the uncertainties by posing more problems than you'll have time to solve. This of course requires more work, but it is especially good when getting started: you'll have more time to prepare for the first session, and you'll have less of an idea of what makes a good problem. Just make sure you start with few enough problems that you can briefly describe them all without taking too much time during the session (and then let people choose what they want to work on). What problems the group decides to work on and makes progress on can also help refine your choice of future problems.

As the group matures, more participants can engage in posing problems. Be sure to communicate what makes a good problem (e.g., by showing them or going over this document), so that problems are more likely to come in a useful form and not waste in-session time (e.g., with background research). If the problem poser is younger in the field, then they probably want to run the problem by a more senior member with more wisdom of experience before posing it publicly.

5 What To Do During an Open Problem Session

This section describes what to do during each open problem session. There are some special tasks for the group's first meeting. Then, during each session, someone needs to give a progress report, someone needs to drive the session, and someone (ideally everyone) needs to take notes. Each of these steps is detailed in the following subsections.

5.1 The First Meeting

Statement of theme. The first meeting should begin with a restatement of the group's theme and the scope of what problems fit in the theme. This was probably already communicated in the invitation, but it's worth restating to allow for discussion and clarification. The scope of the problem session may naturally evolve as the group matures, but the group will have a hard time staying together if it cannot agree on its scope. If there is a list of problems, briefly introducing (all or a sample of) the problems will help concretize the scope.

Statement of rules. The first meeting should also contain a statement of the group's rules about confidentiality and authorship. We follow two rules:

1. **Confidentially:** Problems, discussion, and results are not to be shared with anyone outside the group without the group's permission (provided by either the organizers or everyone involved in the problem).

2. **Authorship:** When it comes time to write the paper (Section 6.2), there will be a call for authors broadcast to the whole group, and each participant *self-determines* whether they are an author, i.e., contributed enough to warrant authorship. Significant contributions can take many forms, from contributing a key idea that worked out, to contributing an idea that didn't work out but kept the problem solving going or inspired another idea, to finding a counterexample to someone else's idea, to writing/working out the details of someone else's idea, to asking key questions that led to an idea getting fleshed out, to posing the problem itself or a helpful subproblem/special case. Authors will then be alphabetical by last name.

These rules are designed to encourage participants to share both their problems and their solution ideas, without fear that they will be stolen without proper credit. By making authorship self-determined and alphabetical, we remove any potential fights over authorship and author order, which in turn helps build group cohesion and a positive atmosphere. Although this model has potential for abuse, we have not seen significant abuse; in fact, we've often found it necessary to convince people that they've contributed enough for authorship when we believe they have. The key is to leave the final decision up to them, and to make the policy clear ahead of time so there are no surprises.

The exact threshold for authorship can vary from person to person, and that's fine. If anyone is unclear on whether their contributions suffice for authorship, they should be encouraged to talk to the organizers. It's also helpful to identify who did what in the notes (Section 5.3) to make this easier to determine.

Social time. Having some informal, non-problem-solving time before the first official meeting can help participants get to know one another. This is especially helpful in groups large enough to split into subgroups working on different problems, as it helps participants find others with similar interests. A standard ice breaker is to get everyone to introduce themselves and their interests.

5.2 Driving

During each open problem session, at least one person needs to be a *driver*. A driver keeps the group organized and on-topic throughout the work session. A driver is a *facilitator* working for the good of the group; their duties may result in them doing less problem-solving work during the meeting, but they enable the other members to be more productive. The driver isn't necessarily the group organizer (though it often is in the beginning), nor need it be the same person from meeting to meeting, and in large groups there can be more than one driver. The driver does not have any authority to decide the direction of the group, but maintains a high-level view of the group, and thus can help it regain direction by reviewing where it stands, trying to find consensus, etc.

At the beginning of each work session, the driver takes the sense of the group as it decides what problem(s) to work on. This may include listing and explaining problems (but keeping it brief), or delegating to the member who posed the problem for an explanation, and answering questions about the problem statements. In groups large enough to split into subgroups, the driver may run an informal interest poll, or assign groups to meet in particular parts of the meeting area ("everyone working on Problem 2, meet in that corner", stated orally or labeling the nearby whiteboard).

One way to run an interest poll (especially at the beginning of a workshop) is to ask participants to upvote (thumbs-up) problems (root messages) they're interested in working on. You can view the results on the group page, reverse sorted by positive emoji.

Especially in small groups, the driver moderates the problem solving discussion. Discussion is a shared medium, so the driver sometimes needs to switch the spotlight and give another contributor time to speak. While having all participants involved in the group discussion at all times maximizes cross-pollination, some participants prefer to think individually (often on paper or a private section of whiteboard) and then share their ideas. The driver needs to pay attention to contributors who have finished working on their own and have something to report to the group. One approach to this is go to around the room and ask people privately what they are thinking about. Often, they have found something interesting, or have a question about an existing idea/approach, but have been too shy to broadcast it.

In larger groups, the driver cannot manage discussion within each subgroup. In this case, the driver's role is to circulate among the subgroups, monitoring each subgroup's progress and

cross-pollinating ideas. For example, if it turns out that two subgroups are working on similar things, the driver can encourage them to merge. Multiple drivers also help in this case.

The driver should also encourage a positive atmosphere of discourse. For example, participants should respect and build on each others ideas (even if that means finding a countereaxmple to disprove an idea) rather than generally disagreeing ("I don't think that's a good idea"). (An analogy is the "Yes, And" principle from improv theater.) We've never seen it, but there's no place for aggression, intimidation, or abrasive behavior in collaborative problem solving.

When a projector is available, the driver usually projects the problem description, any notes from past meetings, and relevant references. This is a useful tool for gently steering the discussion.

✎ In a larger group, when the projector is not otherwise in use, it's helpful to project Coauthor's Live View which shows the latest message(s) that people are working on.

5.3 Taking Notes

Writing down the ideas discussed during a meeting is important for a variety of reasons:

- Having notes available to review during later meetings helps the group pick up where they left off (especially when those meetings were years ago). No matter how long some participants' memories are, they're likely to be exceeded for a long-running group or when writing a paper much later.
- Recording failed approaches helps the group avoid repeating them, and recording unpursued directions reminds the group to come back to them when another approach does not work out.
- Keeping track of progress makes it clear when the results suffice for writing into a paper or other final product.
- Writing down the details of results (e.g., proofs) often uncovers errors in the reasoning, which can then feed back into the discussion. Keeping informal notes can also remind the group of what they had in mind when bugs get discovered later (e.g., when writing the paper).
- Recording which participants participated to each idea helps other participants know who to ask for more information, and make it easier for participants to self-determine authorship later.
- In larger groups, having shared notes makes it easier to see who is working on what, enabling participants to switch over and contribute to other subgroups that interest them.

What to record? Notes should include both partial or complete results (and details/proofs of how the results were obtained), as well as suggested ideas or approaches (even if not explored or shot down), diagrams and drawings, and worked examples. Notes are more detailed than traditional minutes of a meeting, with a focus on preserving reasoning and discussion, not just conclusions. Notes also do not need to follow a chronological format; rather, they should follow the logical order of ideas as that develops.

One easy-to-capture work product that can be included in notes is photographs of whiteboards or scans/photographs of notes on paper. CamScanner is a nice tool for cleaning up such photographs on a mobile device; the images can then be attached to the relevant post on Coauthor. A figure can be worth a thousand words of proof text, and it's generally a lot faster to hand-draw a figure and capture it than to use a drawing program. Even exploratory doodles can be useful

evidence when recalling how the group arrived at its current state of knowledge. The group should get in the habit of photographing all whiteboards before erasing them.

Who takes notes? In small groups, having the driver take notes is convenient, because the driver is always involved in the discussion and may already be projecting notes from a previous meeting. Pausing the discussion to summarize what was just talked about is a natural, useful way for the driver to end a long-running discussion thread and open the floor for a different idea. However, this can also become a bottleneck, and hold the driver back from other driving duties.

A dedicated scribe can also be elected to take notes. When there are multiple subgroups, it's helpful to elect one scribe per subgroup. Putting those notes in a shared medium enables cross-subgroup communication without having to interrupt other subgroups.

Alternately or in addition, all participants should be encouraged to write notes themselves, either during or after the session, about ideas they developed. In our experience, participants need some prodding to actually write notes until they have participated in several meetings, and the other note-taking methods above help illustrate what should be recorded.

Access to notes. Notes should be accessible (and ideally editable) by all participants both in and outside of meetings. Accessible notes allow participants who miss one meeting to keep up with developments, and enable offline writing. Editable notes allow participants to record any ideas that occur to them outside the meeting for further development in the next meeting, and are required for one model of note-taking described above.

Tools for notes. We are actively developing Coauthor, free web software for note-taking, featuring real-time collaborative editing in a hierarchical discussion model (not linear like traditional forums). See https://github.com/edemaine/coauthor/#coauthor for documentation and a more detailed overview of features. Coauthor makes it easy and frictionless to write informal notes, include LaTeX formulas, and attach photographs or other files. The notes are accessible to everyone in the group, but are otherwise private; everyone sees everyone's updates without having to reload the page; and email notifications let people know about updates between sessions.

Many other tools are available, such as wikis, Github, and Dropbox. For drawing figures that fit naturally to a grid, we often use Google Sheets (which makes it easy to add/remove rows/columns and copy/paste rectangles), which can then be turned into paper-quality vector drawings using our tool SVG Tiler (https://github.com/edemaine/svgtiler).

5.4 Progress Reports

Every work session (except the very first) should begin with a progress report summarizing what the group accomplished during the last meeting, any work that occurred between meetings, and the current state of the problem(s) being worked on. Besides just reminding the group where they left off and potentially inspiring new ideas, regular progress reports give the group a chance to evaluate whether the current approach still seem promising or whether they should try a different one, or work on another problem entirely. In large groups, participants may decide to switch groups based on the progress report; for example, they might realize a technique they can apply to someone else's partial progress, or if they previously got stuck on a problem and switched subgroups, they may return to that problem when someone else makes progress. Finally, celebrating progress reinforces the collaborative spirit of the group, especially in large groups where most members are interested in most problems (thanks to the group's shared interest) but cannot keep up with every problem.

✎ The driver should project Coauthor's "Since" view to show all posts since the beginning of the last session.

In large groups working on multiple problems, keeping track of and summarizing progress requires more and more work from the driver, so the driver might ask a member from each subgroup to report progress on that problem. Ideally this is arranged in advance so the progress reporter can prepare.

In small groups, the progress report can naturally segue into work on the problem, but in large groups, the entire group is blocked from splitting into subgroups until they've heard the reports about every problem, so special attention should be paid to keeping each subgroup's progress report short. If a discussion begins during the progress report, the driver should suggest that the discussion participants wait until the progress report is over and the participants can split back into subgroups.

5.5 Reviewing Past Problems

Periodically, the group should reconsider old problems posed but not worked on, or problems with some partial progress but the group got stuck. Like many creative endeavors, having some time away from a problem often leads to new approaches. Solutions to new problems and other work in the field can also provide inspiration. When the group is deciding what to work on at the beginning of a work session, the driver can occasionally ask the group if there are any old problems they would like to revisit. Projecting and flipping through the list of problems can help.

6 What To Do After/Between Sessions

You can help the open problem session run more smoothly with some regular maintenance between work sessions. Also, once your group successfully produces results, you'll need to transition into paper writing. We cover each in the following subsections.

6.1 Maintenance Between Sessions

Maintain notes. You should make sure adequate notes from the problem session have been taken and are properly stored and accessible. Ideally, most of this work will already have been done during the work session itself by whoever was taking notes. But especially early on in a group's lifetime, it's important to check that people actually took sufficient notes and properly distributed them. If details are missing, you can add them yourself or ask the note taker to fill them in. What is sufficient to remember what happened last week may not suffice for six months from now, so especially for important progress, it's good to be thorough. It's also important for note writing to get completed before the next session, in particular as writing can reveal bugs or ideas that may influence the next session.

Review notes. The driver should also look through the notes from the previous work session (and any progress in between sessions) to prepare for the progress report in the next session. Participants should be encouraged to do the same, both to help with the progress report, and to catch up on ideas they may have missed, ensure notes are in order and none of the ideas they recall are missing, and look at the material again with fresh eyes to potentially catch mistakes or ideas that were overlooked during the problem session.

Schedule and confirm meetings. For the weekly meeting format, you'll probably want to send regular reminders about the work sessions (say, the day before), and conversely let everyone know about cancellations because of schedule conflicts. Even if you can make it, the other participants might have schedule conflicts in any given week, so it can be helpful (especially in off times) to find out whether a quorum of people (say, at least three) are around so it makes sense to meet. Even if you can't make it in a given week, other participants may want to meet anyway, in which case you may need to nominate someone as a driver and/or scribe.

Delegating work. You may want to gather volunteers for helping run the open problem session, coming up with an open problem, or otherwise doing work between work sessions. If a driver or note taker is absent a given week, you may need to recruit a backup. During a work session, you may realize that research would go smoother with some supporting work that's best done by one person offline — for example, reading and summarizing a related paper, or developing a software tool to enable better experimentation during a work session. You'll want to solicit for volunteers to do these tasks, and ensure they succeed.

Request feedback. Occasionally, you should check in with the participants to see whether anything can be improved in your open problem session, either in public or one-on-one. For example, are the problems being worked on of interest to the participant? Perhaps problem solving has gone in a different direction than expected and you should re-orient what is being worked on. Do they feel comfortable in the meetings and feel they have an opportunity to contribute? Do they feel OK with how authorship is currently being handled? Are there any interpersonal conflicts that could use a third party to help resolve?

6.2 Writing Papers

Is this a paper? When an open problem session produces enough results, it is time to turn them into papers or other final products. The first question is to decide when the results on a problem are sufficient for being put together into a paper. A related question is when to stop adding new results. There is a natural tension between wanting to finish everything and never finishing, and there are no easy answers. Pay attention to the participants, their desires, and whether progress seems to be slowing.

Call for authors. When a project is ready to turn into a paper, you should send out a call for authors to assemble the list of participants who wish to be authors on a paper (according to the authorship model of Section 5.1). This list may change — for example, someone might prove a new result that makes sense to add, or after further discussion, someone might change their mind — but it's important to have an official current list so that everyone is on the same page. We've found people often have questions about authorship and whether they should be an author, and the organizers or more senior members can help with these judgements (in particular by looking at the notes). In addition, some people may need encouragement if they appear to be making unusual authorship decisions. In many cases, we've had to talk people into being authors on papers they made significant contributions to, but it's important to leave the final decision up to them.

Organize a working group. With the author list formed, you (or the authors) need to decide how to write the paper. People's work styles differ, so different groups might want to call separate meetings to write together, or break the paper into sections and assign them to different people,

or have different people focus on different aspects like prose, figures, and technical proofs. Some authors may want to finish the paper and then pick a venue, while others may want to decide where to send it and use that as a deadline to encourage writing. Basically, at this point you've formed a small separate research group that can manage itself however it prefers. Note, however, that this research group should still keep the rest of the open problem session informed of the progress and any new results, and be open to the possibility that new related results might come from the larger group during the writing process.

Writing. There are many different ways to collaborate on writing a paper. Unfortunately, we've found that meeting as in an open problem session to not be particularly productive. Sometimes it is good for a small group to sit down together to write up challenging pieces of text, but frequently dividing up work and writing on your own seems more effective. Large groups can have diminishing returns in terms of catching errors and structuring words. The authors do need an editing mechanism to ensure unified structure and tone in the writing.

There are many tools to help collaborate on writing a paper. Software like Git and Overleaf can help with simultaneous editing of the document, while maintaining history and backups of past work. Dropbox is a simpler approach, but can often lead to conflicts, which require some care to resolve.

7 Conclusion

We hope you'll try running your own supercollaborative open problem session, and let us about your experience! You can contact us at supercollaboration@csail.mit.edu.