Towards a Characterization of Scheduling Task Complexity

John A. Karasinski* and John Bresina†
NASA Ames Research Center, Moffett Field, CA, USA 94035

Bob Kanefsky‡
San Jose State University, San Jose, CA, USA 95192

Megan Shyr§ and Jessica J. Marquez¶
NASA Ames Research Center, Moffett Field, CA, USA 94035

Future long-duration missions will require astronauts to act more autonomously, manage their schedules, and replan timelines as anomalies and discoveries occur. Astronauts are not professional planners, however, and the complexity of schedules that novice planners can complete successfully is not fully understood. To identify the primary factors which contribute to scheduling task complexity, we conducted a human-in-the-loop study and developed planning algorithms to investigate how the type and amount of constraints affect the difficulty of scheduling and rescheduling. We created rankings of difficulty using a combination of human performance metrics from experimental planning tasks and metrics describing the final plans that participants scheduled. Using the results of our scheduling and rescheduling algorithm algorithms, we created a similar ranking with which to compare. We created rankings which compared well between the experimental and algorithm results for the scheduling task, but the rescheduling task proved more difficult to estimate.

I. Introduction

ASTRONAUTS on the International Space Station currently have their schedules managed by a team of professional planners. Weeks are spent crafting schedules to meet the varied constraints of the science, assembly, and maintenance tasks required to be conducted onboard the vehicle. In the face of emergencies, unscheduled maintenance, or other unplanned events, these planners must reschedule as many tasks as possible while still meeting all the complex constraints between both activities and resources. As NASA considers future long-duration missions to the Moon, Mars, and beyond, research must address the new challenges resulting from the increased communication latency between planners on the ground and crew onboard the vehicle. In these missions, astronauts are envisioned to act with greater autonomy and will need to schedule and reschedule their own timelines. Astronauts are not professional planners, however, and the complexity of schedules that novice planners can complete successfully with regards to performance and cognitive workload is not fully understood. To investigate the primary factors which contribute to scheduling task complexity, we conducted a human-in-the-loop study designed to investigate how the type and amount of constraints affect human performance for scheduling and rescheduling.

Our previous work investigated the various factors around human performance, workload, and situational awareness for an experimental scheduling task [1]. We previously investigated how metrics such as time on task, the number of constraint and overlapping violations, and workload are affected by the different types and amount of constraints. While significant findings were identified (more constraints generally leads to longer time to complete the task and more violations, for instance), these individual metrics did not consistently identify that any one constraint was the most or least challenging to solve. Additionally, previous analyses only investigated the metrics associated with scheduling, and the final plans resulting from the scheduling task have not been investigated. Here, we investigate metrics for measuring the quality of the final plans that participants created in their task, including margin (the amount of unscheduled time in the plan) and the number of unscheduled activities. For a complete discussion of this experiment and the results around the scheduling metrics, see our previous publications [1–3].

*Research AST Human/Machine Systems, NASA Ames HCI Group, AIAA Member.
†AST, Computer Research and Development, Intelligent Systems Division, M/S 269-2.
‡Senior Research Associate, Department of Psychology.
§Pathways Intern, NASA Ames HCI Group, AIAA Member.
¶Human Systems Engineer, NASA Ames HCI Group, AIAA Member.

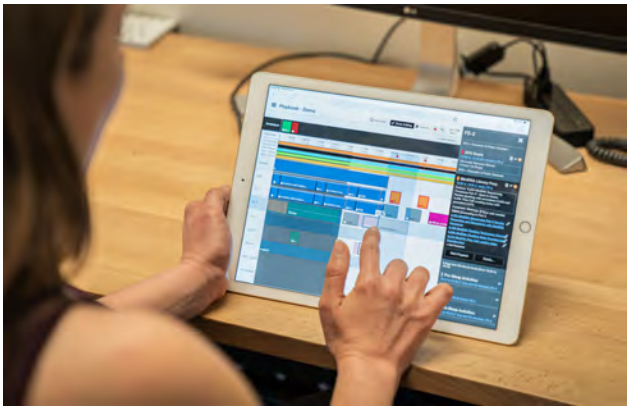


Fig. 1 Playbook is a mobile, web-based scheduling software used to enable crew self-scheduling.

Participants accessed the scheduling platform Playbook [4–6]. Playbook is a mobile, web-based scheduling software used to enable crew self-scheduling, see Fig. 1. In Playbook, the timeline view displays time horizontally from left to right with each crewmember having their own row and their own activities to be executed chronologically. An activity is displayed as a colored rectangle with the length of the block directly proportional to the duration of the activity. Flexible activities (marked with a white dot in the user interface) can be manipulated (i.e., scheduled and assigned by the user), and inflexible activities cannot be moved. An activity may be unconstrained or may have one or more associated constraints. If a constrained activity is scheduled, but the constraint requirements are not met, the activity is marked with a red outline, denoting a constraint-based violation was created. Overlapping activities are also flagged as a violation. For this experiment, all flexible activities had a scheduling priority (high, medium, or low priority). Participants were asked to complete their (re)scheduling efforts by activity priority level — high, medium, and low. Participants were also provided with more activities to schedule than the available time in the timeline, forcing them to make decisions based on this priority. Flexible activities could have associated constraints based on time, resources, equipment, and temporal relation to other activities.

In the field of computer science, problem complexity is characterized into broad classes, e.g., polynomial or exponential, and the complexity class applies to broad problem classes, not to individual problem instances. All of the problem instances in this experiment would fall into the same problem class and, thus, be in the same complexity class. This does not guarantee, however, that human planners will find all scheduling problems similarly difficult to solve. The artificial intelligence (AI) planning field typically uses this problem complexity approach as well [7–10].

While computational approaches for measuring planning complexity have previously been explored [11], they have focused on determining how algorithmically these planning problems can be solved and, in general, only emphasize classes of planning problems. We specifically want to address scheduling task complexity, which may compare one planning problem instance with another, and how that complexity may affect human performance. There has been little research comparing objective measures of scheduling task complexity to human performance measured in controlled laboratory studies [12, 13], though researchers often consider subjective measures of task difficulty [14]. Our aim is to assess the relationship between self-scheduling performance collected through human-subject testing [1] and the objective "difficulty" of the planning problem instance.

We had two primary questions which we aimed to answer in this study:

1) How do different types and amounts of constraints impact human performance in the (re)scheduling tasks?
2) Can we use computational approaches to scheduling difficulty to predict human performance in the (re)scheduling task?

Identifying a robust computational measure of scheduling difficulty will allow us to predict human performance generically and identify scheduling problems that novice planners will be able to solve.

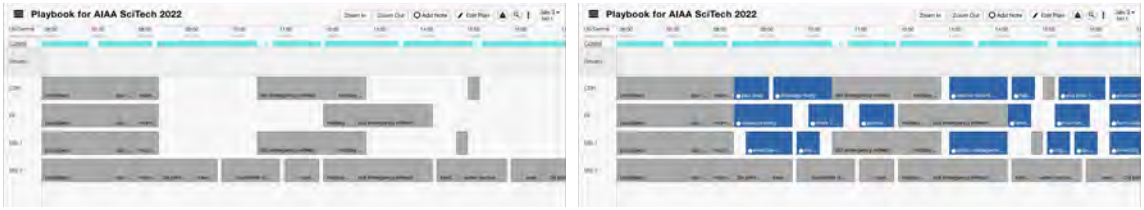
II. Methods

A. Experiment

We recruited 31 participants and split them between two groups based on the task they were asked to complete: scheduling or rescheduling. We developed a baseline schedule similar to those that are used onboard ISS and Earth-based analog missions such as NASA Extreme Environment Mission Operations (NEEMO) [5, 15]. Participants in the scheduling group were presented with an initial timeline that had only pre-planned, inflexible activities, and had several large blocks of time available for the participants to plan activities. The rescheduling group was presented with an initial timeline which contained both inflexible and flexible activities, and were then asked to reschedule as needed to meet their goals (see Fig. 2). Participants were instructed to schedule the activities in order of priority, starting with high, then medium, and finally low. In the experiment, we investigated 4 different types of constraints:

- Time Range Constraint (TR) limits the time of day an activity can be scheduled (e.g., Activity A must start no earlier than 0900 and end no later than 1030);
- Requires Constraint (R) states that the activity needs to have a particular resource available (e.g., Activity A requires communication availability);
- Claim Constraint (CL) describes a specific piece of equipment required for a particular activity (e.g., Activities A and B both claim a treadmill, therefore cannot be scheduled at the same time);
- Ordering Constraint (O) describes when an activity should be scheduled in relation to another activity (e.g., Activity A must be scheduled before Activity B).

In each trial completed by participants, activities had only one type of constraint at one of two levels: low (33% of activities constrained) or high (66% of activities constrained), creating a total of 8 scheduling problems that participants needed to solve. Each participant completed training, a baseline task with no constraints, and the 8 additional tasks which were provided in a random order based on a Latin square design. After completing each trial, participants rated their cognitive workload using the NASA-TLX [16, 17]. For a complete discussion of this experiment and its results, see our previous publications [1–3].



(a) Scheduling (b) Rescheduling

Fig. 2 Examples of the initial conditions for the (a) scheduling and (b) rescheduling trials, showing the partially and fully scheduled timelines, respectively.

B. Modeling

In our previous work, we described an approach for computing the "Expected Solution Quality (ESQ)", which can be used to compare different constrained optimization problem instances [11]. The ESQ approach enables the expression of solution quality and problem-solving computation time using the same metric units; thus, these two factors can be combined into a single metric. The approach involves a statistical characterization of a problem computed from a uniform random sampling of its solution space. For this study, we are using a simplified variation of the ESQ approach that does not require that the random sampling be uniform, and due to this non-uniformity, the solution quality metric and the problem-solving computation metric cannot be combined into a single metric. Furthermore, we are using the size of the search tree as an estimate of problem-solving effort. The margin of the resulting plan, which is the summation of all the time on the plan with no scheduled activities, was also calculated. Plans with a smaller margin are considered to be better than those with a large margin as they make more efficient use of time.

We have developed an iterative random sampling algorithm for each of the two experiment tasks, scheduling and rescheduling. On each iterative sample a solution is incrementally constructed (without any backtracking) by making random choices among the possible options at each problem-solving step. From the results of running these algorithms,

we compute statistics regarding the solution quality as well as regarding the search tree size. Here, the proportional solution quality metric is defined by Equation 1,

$$\mathrm{SQ}_p = 10^2 * \frac{U_H}{T_H} + 10 * \frac{U_M}{T_M} + \frac{U_L}{T_L} \tag{1}$$

where $U_{H,M,L}$ is the number of unscheduled activities with priority high, medium, and low, and $T_{H,M,L}$ is the number of total activities with priority high, medium, and low. The solution quality statistics provide a measure of the difficulty of producing high-quality solutions and the search tree statistics provide a measure of problem-solving effort. These metrics can be used to rank-order the problem instances in terms of "difficulty", in order to compare these two rank orders to participants' performance, as well as their subjective experiences of problem difficulty.

1. Scheduling Algorithm

Participants in the scheduling task were given a partially scheduled timeline (inflexible activities only) and asked to insert new high, medium, and low priority activities. We designed and implemented an iterative sampling algorithm which takes the actions that a human planner takes to solve the scheduling task. Given an initial, partially scheduled timeline that consists of entirely inflexible activities, the algorithm makes random choices in an attempt to schedule all of the new activities in order of priority, see Algorithm 1. In this algorithm, an activity (A) is chosen from the set of activities to be scheduled (H_A, then M_A, then L_A), a valid slot (VS) which the activity can be scheduled in is identified from the set of valid slots (V_S), and the schedule is updated to accommodate the activity. Each V_S is selected from the $OpenLists$ of spaces between inflexible activities, and these $OpenLists$ are updated when new activities are scheduled (after which they are treated as inflexible). V_S are all contiguous unoccupied timespans that are large enough to hold A and in which A can be placed anywhere tip violating any constraints, see Fig. 3. This is evaluated after tentatively applying any moves and removals. Once a VS is chosen, A is scheduled at the beginning (earliest time) in the slot. Activities are selected at random from highest remaining priority until all activities have been scheduled or have failed to be scheduled, and activities that are failed to be scheduled are placed in the set of failed activities (F_A). Once an activity has failed to be scheduled it is no longer considered by the algorithm. The subjects can undo their plan modifications, i.e., backtrack, and the iterative sampling algorithm does not include backtracking; however, the space of possible solutions that can be generated by the subjects is the same as the algorithm's solution space.

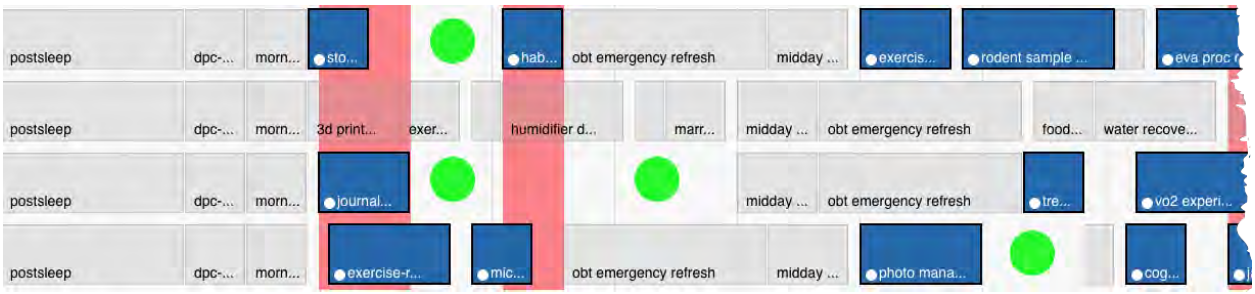


Fig. 3 "Valid Slot" example. The activity to be inserted can fit in any sufficiently wide time slot bounded by other activities already in the same crew member's row (blue or gray) or by keep-out zones (red). In this illustration, the necessary equipment is already claimed (CL) by two already-scheduled activities, and there is a final keep-out zone on the right side stretching to the end of day, due to either a time-of-day (TR) or ordering (O) constraint. (The current experiment did not mix two types of constraints in a single condition.) This leaves four valid slots (green circles).

2. Rescheduling Algorithm

Participants in the rescheduling task were given a fully scheduled timeline (both inflexible and flexible activities) and asked to insert new, high priority activities by removing lower priority activities. As with the scheduling algorithm, we implemented an iterative sampling algorithm which takes the actions that a human planner takes to solve the rescheduling task. Given an initial timeline that consists of flexible and inflexible activities of high, medium, and low priority, the algorithm makes random choices in an attempt to schedule all of the new high priority activities, see

Algorithm 1 Iterative Sampling Algorithm for Scheduling

```
Initialize Schedule, OpenLists
H_A, M_A, L_A ← Sets of Prioritized Activities
F_A ← {}
TreeSize ← 1
for S_A in {H_A, M_A, L_A} do
    for A in S_A do
        V_S ← Valid slots in OpenLists
        if V_S is { } then
            Remove A from S_A
            Add A to F_A
        else
            TreeSize *= size(V_S)
            VS ← random choice from V_S
            procedure Schedule
                Update Schedule to reflect insertion of A
                Update OpenList to reflect insertion of A
                Remove A from S_A
            end procedure
        end if
    end for
end for
```

Algorithm 2 Iterative Sampling Algorithm for Rescheduling

```
Initialize Schedule, OpenLists
H_A ← Set of New High Priority activities
M_A, L_A ← {}
F_A ← {}
TreeSize ← 1
for S_A in {H_A, M_A, L_A} do
    for A in S_A do
        V_S ← Valid slots in OpenLists
        if V_S is { } then
            Remove A from S_A
            Add A to F_A
        else
            TreeSize *= size(V_S)
            VS ← random choice from V_S
            Compute C_S from VS
            Reschedule ← random choice from C_S
            TreeSize *= size(C_S)
            procedure Reschedule
                Add each removed activity to H_A, M_A, L_A
                Update Schedule to reflect each activity removal
                Update Schedule to reflect insertion of A
                Update OpenList to reflect insertion of A
                Remove A from S_A
            end procedure
        end if
    end for
end for
```

Algorithm 2. As in Algorithm 1, an activity (A) is chosen from the set of new high priority activities to be scheduled (H_A), a valid slot (VS) which the activity can be scheduled in is identified from the set of valid slots (V_S), and the schedule is updated to accommodate the activity. Here, V_S are defined such that, if all the flexible activities were removed, there is enough room (with no overlaps) to insert A without violating any associated constraints. The core of this rescheduling algorithm involves creating a candidate set (C_S) of operations to insert the new high priority activities by either moving or removing the activities that have been currently scheduled (these operations are further described below). A random choice from this set of candidate operations is then chosen until all activities (including the set of medium, M_A, and low, L_A priority activities that have been removed to accommodate the new high priority activities) have been scheduled or have failed to be scheduled. Activities that are failed to be scheduled are again placed in the set of failed activities (F_A) and no longer considered by the algorithm.

Each C_S consists of two types of insertions that approximate the actions that a planner takes during their rescheduling task. Both of these actions attempt to insert Activity A with duration D into the schedule. The two types of actions are to

1. Insert an activity by moving already scheduled activities (without removing activities). To do this, the algorithm
    1.1. Considers the windows between two consecutive fixed activities on the schedule. If a selected window has a total margin greater than or equal to D, then continue. See Fig. 4 for an example.
    1.2. Determines if A can be inserted between the fixed activity at the left side of the window and first flexible activity in the schedule, between every pair of flexible activities, and between the last flexible activity and the fixed activity at the right side of the window.
    1.3. For each of these candidate insertions, move any activities before the insertion point as far left as possible within the window (respecting any constraints), and moves any activities after the insertion point as far right as possible. If the gap produced by these moves has a span greater or equal to D, then the start time of this gap is a valid choice to add to the set of candidate choices.
2. Insert an activity by removing already scheduled activities. To do this, the algorithm
    2.1. Considers all sequential subsets of the flexible activities for removal. First the removal of single activities is considered, then removal of consecutive pairs of activities are considered, then consecutive triples of activities are considered, etc. Then the constraints of the remaining activities must be applied, which creates "keep-out zones."
    2.2. For each removal option:
        2.2.1. If there is room to fit the task into a valid interval (between keep-out zones) then it is placed as early as possible.
        2.2.2. If there is no room, then apply the procedure from 1.3. to move flexible activities to enable insertion and add candidates to the set.

To build the C_S, the algorithm identifies all valid actions of these two types.

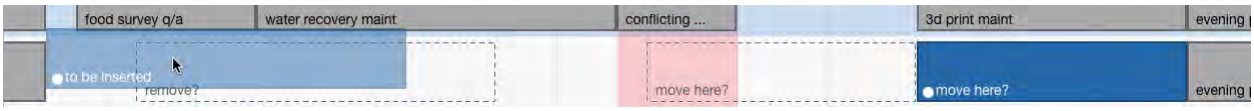


Fig. 4 Example of one option for inserting an activity in a crew member's row. Two flanking inflexible activities (gray) have been selected. It must avoid overlapping any other activities (blue) in the selected row or any area that would violate a constraint (red), in this case a conflicting claim by an activity in another crew member's row. This particular option involves removing one activity and moving another as far right as its own constraints allow.

III. Results

A. Experiment

Instead of only investigating metrics around the scheduling task itself, our current work examines the metrics associated with the final plans that participants created. Here we investigated margin and proportional solution quality (SQ_p). Unlike the metrics based around the scheduling task, these plan metrics can be directly compared between the experimental and modeling efforts. While we did not expect the algorithm results to be directly comparable to those from the experiment, as the algorithm is based around making random choices rather than the purposeful decisions the

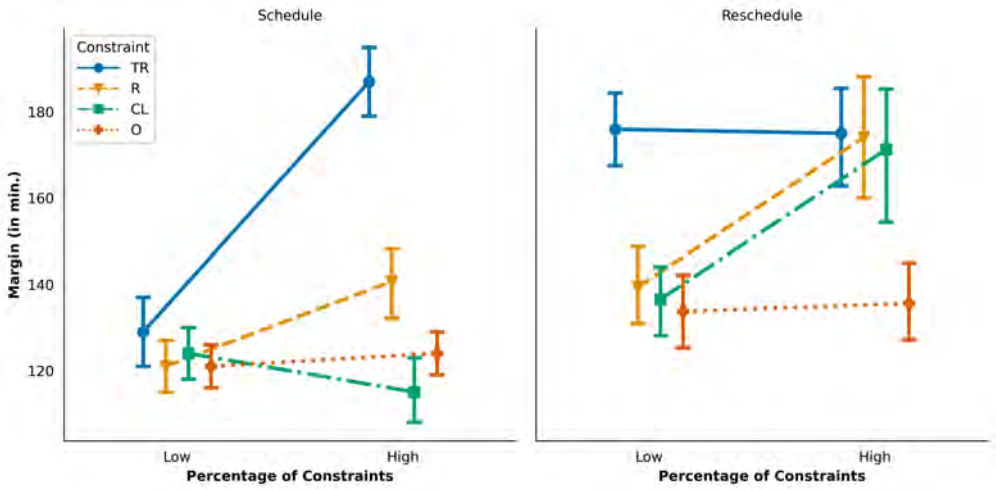


Fig. 5 The margin remaining from the experiment across the two types of task, four types of constraints, and two amounts of constraints. Error bars represent the standard error of the mean.

participants made, we analyzed the metrics to rank the different conditions in order of difficulty.

For both of our human performance metrics, we used linear mixed effects models with one between-participants variable (type of task, with 2 levels) and two between-participants variables (type of constraint, with 4 levels; and number of constraints, with 2 levels). Participants were added into these models as a random factor. When statistically significant effects were identified, post-hoc comparison tests were conducted using Tukey Honest Significance Tests with a Bonferroni correction. When necessary due to missing data, Satterthwaite's method was used to adjust the degrees of freedom.

Analysis of the margin of participants' final plans indicates that there were significant main effects due to the type of task ($F(3, 29.04) = 6.01, p = 0.02$), type of constraint ($F(3, 87.17) = 18.30, p < 0.001$), and number of constraints ($F(3, 116.02) = 25.66, p < 0.001$). There was also a significant two-way interaction effect between type and number of constraints ($F(3, 116.02) = 3.17, p = 0.03$), and a significant three-way interaction effect between type of task, type of constraint, and number of constraint ($F(3, 116.02) = 9.47, p < 0.001$). This three-way interaction effect reflects that there were roughly two margin sizes that tended to remain at the end of the experimental trials. Participants were instructed to attempt to minimize the margin in their final plans, and a lower margin indicates that participants planned more efficiently. Within the different constraint types, the planning instance was exactly the same, but they resulted in very different margins across the type of task participants were completing. In the scheduling task, participants tended to always have roughly the same amount of margin remaining at the end of their trials, except for TR_{High}, which resulted in a significantly higher margin. For the rescheduling task, participants tended to again have the most difficulty scheduling trials with the TR constraints. The TR and O constraints resulted in roughly the same amount of margin remaining at the end planning process, regardless of the number of activities with constraints. In contrast, the R and CL constraints had significantly more margin in the high number of constraints conditions than the low number of constraints trials. See Fig. 5 for a breakdown of the remaining margin in the participants' plans.

Analysis of the solution quality identified a significant main effect due to the type of constraint ($F(3, 173.52) = 3.29, p = 0.02$). The type of task and amount of constraints were not significantly different. Post-hoc pairwise comparisons show that this effect is largely driven by the CL constraints. These trials had significantly higher solution quality (i.e., worse resulting plans) than O ($p = 0.02$), and the CL trials tended to have the equivalent of one additional medium priority activity unscheduled. Given the constraints of the inflexible activities initially in the plan, it was not possible to schedule all activities, though it was possible to only leave 2 activities unscheduled (resulting in a solution quality of 0.250). It should be noted, however, that the vast majority of participant trials (226 of 248) had a solution quality ≤ 0.75, indicating that participants had scheduled all high and medium priority activities and had 6 or less low priority activities unscheduled. Roughly 95% of scheduling trials and 88% of rescheduling trials had a $\text{SQ}_p < 1$. Additionally, 194 of 248 trials resulted in a solution quality of either 0.375 or 0.500, indicating that participants normally scheduled all but 3 or 4 activities, respectively.

B. Modeling

We ran 10,000 samples in each condition for a total of 180,000 simulations for this analysis. On each iterative sample a solution was incrementally constructed by making random choices among the possible options at each problem-solving step. The result of these random choices are sets of plans which are, on average, much worse than the intentional plans created by our novice planners. Despite this, both of the algorithms were designed based on the strategies employed by the participants in the experiment and were capable of finding the same solutions that participants did.

The minimum solution quality possible on all trials was 0.250, which was found in 7 of the 18 conditions. (Only one of these conditions, the baseline, occurred in the rescheduling results.) Additionally, 7 of the 18 conditions found solutions with only 3 activities remaining. The scheduling algorithm performed notably better than the rescheduling algorithm, however, resulting in plans with much lower scheduling quality. Based on how the rescheduling algorithm was designed, though, this is not surprising. Considering the random nature of the choices made at each step, it was much easier for the scheduling algorithm to make "lucky" random choices. If the rescheduling algorithm failed to reschedule a single high priority activity early in the planning process, it could never improve that iteration's solution quality below 100. At least one high priority activity remained in the final plan much more frequently for rescheduling algorithm results (75,328 of 90,000 samples) than the scheduling algorithm results (5,830 of 90,000 samples).

Due to the nature of the solution quality metric, the resulting distributions were not normal and median values were used for the rest of this analysis. The resulting median solution quality of the samples varied greatly depending on the type and number of constraints. For the scheduling task, all the constraints except for R showed an increase in solution quality as the number of constraints increased from Low to High. This difference is reflective of roughly one additional medium priority activity being unscheduled in the High constraint conditions. For the rescheduling task, the O and R constraints showed a similar trend as the number of constraints increased. TR remained flat, however, while CL showed a large increase when constraints increased. Complete results of the solution quality are available in Table 1.

C. Rank Comparisons

To compare between our experimental and modeling rankings, we first determined which metrics to include. We included measures of efficiency, accuracy, and workload in order to capture the "difficulty" of the experimental trials. These scheduling metrics included time on task, the number of constraint based and overlapping violations made while planning, and the participants' workload. We then also included scheduling quality and the margin of the final plans that participants created. We separated the data between the two groups of participants, scheduling and rescheduling, as the differences between the two tasks did not allow for direct comparisons. These metrics were then converted to z-scores, representing the number of standard deviations by which the value of a raw score is above or below the mean value of the metric. Finally, the z-score transformed metrics were averaged and summed for each condition to create a final score. The four scheduling metrics were each given a weight of 0.25, while the two plan metrics were given a weight of 0.50 before summing, such that both aspects of the task had equal weight.

To create the rankings from the modeling data, we initially investigated tree size, scheduling quality, and margin. Tree size, which relates to the problem-solving computation time was ultimately removed, however, as it wasn't found to be correlated with any of the experimental metrics. The results for minimum, mean, and median tree size are available in Table 2. For the final modeling rankings, the scheduling quality and margin were put through a similar procedure as the experimental metrics. Both metrics were split between the two tasks, transformed to z-scores, and equally weighted to create the final rankings.

For the scheduling experiment, we found that the hardest problems to solve generally had a higher number of constraints, and that each constraint's high number condition was ranked more difficult than the corresponding low number condition. The experimental trials identified TR, CL, and O as the most challenging, while R and the Baseline trial were ranked as easier. These results generally show good agreement with the rankings produced by the scheduling algorithm, which also identified TR, O, and CL as the top three hardest problems. While R and the Baseline were also identified as the easiest problems, one quirk of the scheduling algorithm rankings is that R_{High} was ranked as being easier to solve than R_{Low}. This quirk appears to be a result of the scheduling algorithm producing the smallest margin compared to all the other conditions, which was not found in the experimental trials. Aside from this constraint type, however, we again found that all High number of constraint conditions ranked as more difficult than their Low number of constraints counterparts.

For the rescheduling experiment, we found that the CL_{High} and TR_{High} conditions ranked the hardest, while R_{Low} and O_{Low} ranked easiest. The algorithm, on other hand, identified O_{High} and CL_{High} as as the hardest, while R_{Low} and TR_{High} ranked easiest. A notable difference between the experimental and algorithm results for rescheduling is

Table 1 Solution Quality

Task	Constraint	Percentage	Min	Mean	Median
Schedule	CL	High	0.375	1.225	1.500
		Low	0.500	0.805	0.500
	O	High	0.250	5.897	1.625
		Low	0.250	3.000	0.500
	R	High	0.250	0.435	0.375
		Low	0.250	0.546	0.500
	TR	High	0.375	2.228	1.875
		Low	0.250	0.834	0.500
	Baseline	—	0.250	0.944	0.500
Reschedule	CL	High	0.375	22.829	26.375
		Low	0.375	21.631	17.688
	O	High	0.375	29.736	27.875
		Low	0.625	24.949	26.625
	R	High	0.625	19.177	16.500
		Low	0.375	17.006	15.375
	TR	High	0.750	15.332	15.375
		Low	0.375	17.808	15.500
	Baseline	—	0.250	17.637	15.375

Table 2 Tree Size

Task	Constraint	Percentage	Min	Mean	Median
Schedule	CL	High	5E+03	8E+05	3E+05
		Low	1E+04	5E+05	3E+05
	O	High	9E+07	2E+11	5E+10
		Low	7E+08	2E+12	6E+11
	R	High	3E+10	3E+13	1E+13
		Low	5E+10	3E+13	1E+13
	TR	High	1E+06	8E+07	4E+07
		Low	1E+08	1E+11	5E+10
	Baseline	—	1E+04	4E+05	2E+05
Reschedule	CL	High	2E+06	4E+10	2E+09
		Low	1E+06	2E+11	3E+09
	O	High	2E+06	1E+10	5E+08
		Low	2E+06	1E+10	1E+09
	R	High	5E+06	5E+11	3E+10
		Low	3E+06	5E+11	1E+10
	TR	High	4E+03	2E+05	9E+04
		Low	6E+04	4E+08	4E+07
	Baseline	—	5E+06	1E+11	6E+09

Table 3 Scheduling Problem Difficulty Rankings, from Hard to Easy

(a) Experimental Data

Constraint	Percentage	Rank	Score
TR	High	1	1.36
CL	High	2	0.26
O	High	3	0.11
CL	Low	4	-0.06
R	High	5	-0.07
TR	Low	6	-0.08
O	Low	7	-0.34
Baseline	—	8	-0.41
R	Low	9	-0.68

(b) Algorithm Result

Constraint	Percentage	Rank	Score
TR	High	1	0.67
O	High	2	0.38
CL	High	3	0.25
CL	Low	4	0.15
O	Low	5	-0.12
TR	Low	6	-0.24
Baseline	—	7	-0.25
R	Low	8	-0.25
R	High	9	-0.59

Table 4 Rescheduling Problem Difficulty Rankings, from Hard to Easy

(a) Experimental Data

Constraint	Percentage	Rank	Score
CL	High	1	1.13
TR	High	2	0.71
TR	Low	3	0.30
R	High	4	0.19
CL	Low	5	-0.21
O	High	6	-0.31
Baseline	—	7	-0.48
R	Low	8	-0.57
O	Low	9	-0.75

(b) Algorithm Result

Constraint	Percentage	Rank	Score
O	High	1	0.40
CL	High	2	0.14
O	Low	3	0.07
CL	Low	4	0.06
R	High	5	0.03
TR	Low	6	-0.13
Baseline	—	7	-0.15
R	Low	8	-0.15
TR	High	9	-0.27

that the O constraint, which ranked relatively easy for the participants, was identified as the most challenging for the algorithm. This is the only constraint type that required two activities to be scheduled together, while all the other constraint types required the opposite. Participants most likely rescheduled the two activities consecutively, whereas the order of activities rescheduled by the algorithm is random—no dependencies (via constraints) are considered. In fact, both algorithms show the highest average solution quality for the four O constraint conditions, suggesting that the solution quality had a large negative skew for this constraint type. In general, the rankings for the rescheduling problem do not align as well as those for the scheduling algorithm. The final rankings for the scheduling and rescheduling problems are available in Tables 3 and 4.

IV. Discussion

Future long-duration missions will require astronauts to act more autonomously, manage their schedules, and replan as anomalies and discoveries occur. Astronauts are not expert planners, however, and software aids can be designed to help support the planning task. To know where support countermeasures are needed, we must first rank scheduling and rescheduling problems to identify which constraints cause the most difficulty. Additionally, being able to assess a timeline for its scheduling task complexity would help expert planners determine if astronauts could (re)schedule that particular day. We created rankings of difficulty using a combination of human performance metrics from the experimental planning tasks and metrics describing the final plans that participants scheduled. Using the results of our

scheduling and rescheduling algorithm simulations, we created a similar ranking with which to compare, in the hopes of being able to predict the relative difficulty of future planning tasks. While we were able to create rankings which compared well between the experimental and algorithm results for the scheduling task, the rescheduling task proved more difficult to estimate. This may simply follow from the poor solution quality generally found by the rescheduling algorithm–indicating that it does not perform as well as the scheduling algorithm. The rescheduling task has more types of plan modification operators, and it seems that there are more "bad" choices that impact future choices (due to not allowing backtracking, and only allowing an activity to be attempted to be rescheduled only once). These factors could result in worse median quality scores and might require higher numbers of samples to find the best solutions. The strategy taken by the rescheduling algorithm—to attempt to schedule high priority, then medium, then low—may not be reflective of the participants actual behavior. Future work can aim to better identify the strategies that the rescheduling participants were taking and attempt to integrate these into a future rescheduling algorithm.

Statistical analysis of the human performance metrics suggests that the TR constraint caused participants to create plans with significantly more margin than the other constraint types, and identified that the CL constraint resulted in plans with the worst solution quality. This held true across both the scheduling and rescheduling tasks, and the algorithm also consistently identified CL in the upper half of the difficult problems. The TR and CL constraint types seem to be significantly more difficult for participants to schedule (and reschedule) compared to the other types of constraints, especially when there are many of these constraints that need to be resolved. When considering future scheduling aids for novice planners, it may be useful to prioritize new features that help manage these constraints specifically. Alternatively, when considering future long-duration missions, it may be that plans with many of these constraints need to be handled by expert planners on Earth rather than allowing astronauts onboard to resolve that part of the planning process.

One limitation of this study was that each constraint was only considered by itself, and that no experimental or modeling trials included constraints of multiple types. It is not clear how different combinations and types of constraints would interact with each other. Ongoing research efforts at NASA's Human Exploration Research Analog (HERA) Campaign 6 are tasking analog crews with scheduling their operational timeline. These operational timelines include a varied number and type of constraints more similar to those that astronauts may face on future long-duration exploration missions. While none of the modeling results in this present study investigated multiple constraint types, the algorithms themselves are already capable of scheduling with varied constraints. Future work could investigate the timelines that analog crew in HERA C6 are currently being asked to schedule to explore if the rankings developed here are applicable to operational timelines.

Acknowledgments

This research was funded by the NASA Human Research Program's Human Factors and Behavior Performance Element (NASA Program Announcement number 80JSC017N0001-BPBA) Human Capabilities Assessment for Autonomous Missions (HCAAM) Virtual NASA Specialized Center of Research (VNSCOR) effort.

References

[1] Marquez, J. J., Edwards, T., Karasinski, J., Lee, C., Shyr, M., Miller, C., and Brandt, S. L., "Human Performance of Novice Schedulers for Complex Spaceflight Operations Timelines," *Human Factors*, in press.

[2] Edwards, T., Brandt, S. L., and Marquez, J. J., "Towards a Measure of Situation Awareness for Space Mission Schedulers," *Advances in Neuroergonomics and Cognitive Engineering*, Vol. 259, edited by H. Ayaz, U. Asgher, and L. Paletta, Springer International Publishing, Cham, 2021, pp. 39–45. https://doi.org/10.1007/978-3-030-80285-1_5, URL https://link.springer.com/10.1007/978-3-030-80285-1_5.

[3] Shyr, M., Edwards, T., Brandt, S. L., and Marquez, J. J., "The Path to Crew Autonomy - Situational Awareness in Scheduling and Rescheduling Tasks for Novice Schedulers," *72ND INTERNATIONAL ASTRONAUTICAL CONGRESS*, Virtual - Dubai, 2021.

[4] Marquez, J. J., Pyrzak, G., Hashemi, S., McMillin, K., and Medwid, J., "Supporting Real-Time Operations and Execution through Timeline and Scheduling Aids," *43rd International Conference on Environmental Systems*, American Institute of Aeronautics and Astronautics, Vail, CO, 2013. https://doi.org/10.2514/6.2013-3519.

[5] Marquez, J. J., Hillenius, S., Kanefsky, B., Zheng, J., Deliz, I., and Reagan, M., "Increasing crew autonomy for long duration

exploration missions: Self-scheduling," *2017 IEEE Aerospace Conference*, IEEE, Big Sky, MT, USA, 2017, pp. 1–10. https://doi.org/10.1109/AERO.2017.7943838.

[6] Marquez, J. J., Hillenius, S., Zheng, J., Deliz, I., Kanefsky, B., and Gale, J., "Designing for Astronaut-Centric Planning and Scheduling Aids," *Proceedings of the Human Factors and Ergonomics Society Annual Meeting*, Vol. 63, No. 1, 2019, pp. 468–469. https://doi.org/10.1177/1071181319631386.

[7] Bylander, T., "Complexity results for planning," *Proceedings of the 12th International Joint Conference on Artificial Intelligence - Volume 1*, Morgan Kaufmann Publishers Inc., Sydney, New South Wales, Australia, 1991, pp. 274–279.

[8] Erol, K., Hendler, J., and Nau, D. S., "HTN planning: complexity and expressivity," *Proceedings of the Twelfth AAAI National Conference on Artificial Intelligence*, AAAI Press, Seattle, Washington, 1994, pp. 1123–1128.

[9] Erol, K., Nau, D. S., and Subrahmanian, V., "Complexity, decidability and undecidability results for domain-independent planning," *Artificial Intelligence*, Vol. 76, No. 1-2, 1995, pp. 75–88. https://doi.org/10.1016/0004-3702(94)00080-K.

[10] Bäckström, C., and Nebel, B., "COMPLEXITY RESULTS FOR SAS $^+$ PLANNING," *Computational Intelligence*, Vol. 11, No. 4, 1995, pp. 625–655. https://doi.org/10.1111/j.1467-8640.1995.tb00052.x.

[11] Bresina, J., Drummond, M., and Swanson, K., "Expected solution quality," *Proceedings of the Fourteenth International Joint Conference on Artificial Intelligence*, Morgan Kaufmann Publishers Inc., Montreal, Quebec, Canada, 1995, pp. 1583–1590.

[12] Maynard, D. C., and Hakel, M. D., "Effects of Objective and Subjective Task Complexity on Performance," *Human Performance*, Vol. 10, No. 4, 1997, pp. 303–330. https://doi.org/10.1207/s15327043hup1004_1.

[13] Moray, N., Dessouky, M. I., Kijowski, B. A., and Adapathya, R., "Strategic Behavior, Workload, and Performance in Task Scheduling," *Human Factors: The Journal of the Human Factors and Ergonomics Society*, Vol. 33, No. 6, 1991, pp. 607–629. https://doi.org/10.1177/001872089103300602.

[14] Mangos, P. M., and Steele-Johnson, D., "The Role of Subjective Task Complexity in Goal Orientation, Self-Efficacy, and Performance Relations," *Human Performance*, Vol. 14, No. 2, 2001, pp. 169–185. https://doi.org/10.1207/S15327043HUP1402_03.

[15] Lee, C., Marquez, J., and Edwards, T., "Crew Autonomy through Self-Scheduling: Scheduling Performance Pilot Study," *AIAA SciTech 2021 Forum*, American Institute of Aeronautics and Astronautics, VIRTUAL EVENT, 2021. https://doi.org/10.2514/6.2021-1578.

[16] Hart, S. G., and Staveland, L. E., "Development of NASA-TLX (Task Load Index): Results of Empirical and Theoretical Research," *Advances in Psychology*, Vol. 52, Elsevier, 1988, pp. 139–183. https://doi.org/10.1016/S0166-4115(08)62386-9.

[17] Hart, S. G., "NASA-Task Load Index (NASA-TLX); 20 Years Later," *Proceedings of the Human Factors and Ergonomics Society Annual Meeting*, Vol. 50, No. 9, 2006, pp. 904–908. https://doi.org/10.1177/154193120605000909.