Action Schema Neural Networks: Generalized Policies for Stochastic Planning Problems In The Wargaming Domain

Ronald Picard
Vanderbilt University
Nashville, TN, USA

***Abstract*—Stochastic shortest path problems have been of interest to the automated planning community for many years. Traditionally, policy solutions to these problems have been found by a set of admissible heuristics, such as LM-Cut, which are able to approximate the best actions to take in a current state to provide the highest probability of reaching a goal state in a delete relaxation of these problems. Though successful, these heuristics face scalability problems as the state spaces of these stochastic problems increase. [1] provided a solution to this problem by utilizing deep neural networks to learn a successful policy that could scale to Nth order problems with only linear time constraints. The neural networks are coined Action Schema Networks (ASNets), since given a current state they provide an appropriate action to take. We present a case study on this technique by applying it to the fighter jet wargaminig domain. We have designed a PPDDL domain and grounding files for a wide set of scenarios in which red and blue, 4th and 5th, generations fighters engage in battle and the ASNets must decide on which attack method to use given the current scenario state to increase probability of reaching a goal state. We present the results of 5 trial experiments and discuss the degree of success we have had in training the ASNets, intuition about the results, and suggestions for future work.**

***Index Terms*—Artificial Intelligence, Machine Learning, Neural Networks, ASNets, Tesorflow, Keras, Stochastic Shortest Path Problems, Action Schema Networks, PDDL, PPDDL, Automated Planning, LM-Cut**

I. Introduction

Stochastic shortest path (SSP) problems have been of interest to the planning community for many years. These problems involve stochastic transitions between states, and provide a challenge for planning problems in which we wish to maximize the probability of reaching a desired goal state from an initial state. Because the transitions are probabilistic in nature, there is usually no guarantee that a goal state will be reached; however, there are many delete relaxation heuristics that will provide successful solutions. However; scalablility of these heuristics is often leads to long solution times. The need for generalized policies that can provide successful solutions to a generalized set of SSP problems in an expeditious manner is prevalent. The work of [1] provides the foundations of training neural networks (NNs) to find generalized policies that are applicable to all SSPs of a particular domain. These networks are coined ASNets. We present a case study in which ASNets are applied to the aircraft wargaming domain. In this domain specific actions must be taken to provide high statistical probability of success; namely, that all red aircraft will be destroyed.

II. Problem Background

In this section we introduce some fundamental topics required to understand the different sections of this work.

A. Stochastic Shortest Path Problems

Formally, a stochastic shortest path problem (SSP) is a tuple (S, A, T, C, G, so). S is a finite set of states, A is a finite set of actions, $T : SxAxS \rightarrow [0, 1]$ is a transition function, $C : SxA \rightarrow (0, \infty)$ is a cost function, G is a set of goals states, and s_0 is an initial state. [1] According to [1], the solution of an SSP is a policy $\pi : AxS \rightarrow [0, 1]$ such that $\pi(a|s)$ is the probability that action will be applied in state s. $\pi*$ represents the policy with the minimum cost to reach to the goal.

B. LM-Cut Heuristic

The Landmark Cut Heuristic (LM-Cut), $h^{LM-Cut}(G)$, is a heuristic used to approximate the optimal cost solution, $h^+(G)$, to the delete relaxed versions of planning problems. A delete relaxed planning problem is a problem in which the deletion requirement of a post condition is relaxed (removed), thus making the problem easier to solve. LM-Cut does this by identifying disjunctive action landmarks, which are sets of actions in which at least one action in each landmark is required to be taken to reach the goal in the relaxed problem. This heuristic is admissible meaning that it never overestimates the optimal cost of reaching the goal; or more formally $h^{LM-Cut}(G) \leq h^+(G)$. [2] The following procedure for the LM-Cut heuristic is provided by [3].

1) Initialize $h^{LM-Cut}(I) := 0$ Then iterate:
2) Compute h^{max} values of all variables
3) Let P be a precondition function (pcf) that chooses the precondition with maximal h^{max} values
4) Compute a cut which guarantees $cost(L) > 0$
5) Increase $h^{LM-Cut}(I)$ by $cost(L)$
6) Decrease cost(o) by cost(L) for all $o \in L$
7) Compute a cut which guarantees $cost(L) > 0$ for the corresponding landmark L as follows:
   a) The goal zone V_g of the justification graph consists of all nodes that have a path to g where all edges are labelled with zero-cost operators.

b) The cut contains all edges $\langle v, o, v' \rangle$ such that $v \in V_g$, $v' \in V_g$ and v can be reached from i without traversing a node in V_g

C. Action Schema Networks

Action Schema Networks (ASNets) are supervised deep neural networks applicable SSPs. The goal is to train these networks to learn a policy for selecting actions based on current states that will lead to the highest probability of reaching the goal state. In this work, LM-Cut is used at the teacher heuristic that will provide the appropriate true labels (action given a state) to the ASNets.

D. Planning Domain Definition Language

Planning Domain Definition Language (PDDL) is a declarative description language used as a standard for describing artificial intelligence planning problems. The language allows for the specification of a problem domain, then a set of groundings (instantiations) to describe specific scenarios.

1) Domain: The domain of a PDDL describes generic object-types, predicates over those object-types, and actions with a set of preconditions and post conditions.

2) Groundings: A grounding of a PDDL domain is an instantiation of the domain in which objects of the generic types of the domain are instantiated, with a set of initial conditions (predicates over those objects) based on the predicates of the domain, and a goal state specification with reward metrics.

E. Probabilistic Planning Domain Definition Language

Probabilistic Planning Domain Definition Language (PPDDL) is an extension to PDDL in which effects (post conditions) of an action may take place with a certain probability. This allows for the specification of SSP problems within the PDDL language.

F. Planning Problem

The planning problem designed for this work is a wargame between red and blue fighter jet teams. In each problem scenario both the red and blue teams each have N number of fighters. The fighters of each team are made up of random variations of fourth-gen and fifth-gen fighters. There does not have to be an equivalent number of fighters from a specific generation on both teams; but rather, the generation can vary. However, the number of fighters per team is equivalent and fixed. The N number of fighters on each team are ordered in a line. During each round, the front two fighters fight providing four possible engagements types during each round: fourth-on-fourth, fifth-on-fifth, fourth-on-fifth, and fifth-on-fourth. During each round; one of the two front fighters is destroyed and the other survives and maintains it front position status. The fighter that is destroyed, is then replaced with next fighter in line. If the destroyed fighter was the last fighter alive then an end state is reached. During each round the blue fighter must select from one of three strike actions; strike-short-range, strike-long-range, and strike-stealth. Each action will result in a probabilistic outcome based which of the four types of engagements is currently happening. The goal of the planning problem is to select the best action based on the current state (type of engagement and whether the blue aircraft is the last alive) that will result in the highest probability of reaching the goal state. The goal state is that all of the red team fighters are destroyed and at least one blue team fighter is not destroyed. A dead end state is reached if all blue team fighters are destroyed. The following list provides a list of actions with probabilities of success based on the current state. Specific actions are best for every state.

1) short-range-strike
   a) fourth-on-fourth: 0.6 blue wins, 0.4 red wins
   b) fifth-on-fifth: 0.4 blue wins, 0.6 red wins
   c) fifth-on-fourth: 0.7 blue wins, 0.3 red wins
   d) fourth-on-fifth: 0.2 blue wins, 0.8 red wins
2) long-range-strike
   a) fourth-on-fourth: 0.4 blue wins, 0.6 red wins
   b) fifth-on-fifth: 0.6 blue wins, 0.4 red wins
   c) fifth-on-fourth: 0.8 blue wins, 0.2 red wins
   d) fourth-on-fifth: 0.3 blue wins, 0.7 red wins
3) stealth-strike
   a) fourth-on-fourth (not last): 0.7 blue wins, 0.3 red wins
   b) fourth-on-fourth (last): 0.3 blue wins, 0.7 red wins
   c) fifth-on-fifth (not last): 0.7 blue wins, 0.3 red wins
   d) fifth-on-fifth (last): 0.3 blue wins, 0.7 red wins
   e) fifth-on-fourth (not last): 0.9 blue wins, 0.1 red wins
   f) fifth-on-fourth (last): 0.7 blue wins, 0.3 red wins
   g) fourth-on-fifth (not last): 0.3 blue wins, 0.7 red wins
   h) fourth-on-fifth (last): 0.1 blue wins, 0.9 red wins

III. Implementation

A. PPDDL

A set of PPDDL files; both a domain, and grounding files are developed of this work. A python generator script and bash generator script are developed to auto generate large sets of grounding files for this PPDDL domain.

1) Domain: The PPDDL wargame domain for this work is made up of four sections: flags, types, predicates, and actions. The flags are: requirements, probabilistic-effects, conditional-effects, equality, typing, and rewards. There is only one type; fighter. The predicates are blue, red, fourth-gen, fifth-gen, front, last, destroyed, and behind. There are three actions specified for this domain; strike-short-range, strike-long-range, and strike-stealth. Each action takes placed based on the same preconditions listed below. The effect of each action is probabilistic; with the outcome depending on which generations are the front fighters in the current state. In every outcome, one fighter is destroyed and other team's survive. With this the destroyed fighter is replaced with the fighter behind it (if there is one) to take the next action.

1) predicates:
   a) (blue ?f - fighter)
   b) (red ?f - fighter)
   c) (fourth-gen ?f - fighter)

d) (fifth-gen ?f - fighter)
e) (front ?f - fighter)
f) (last ?f - fighter)
g) (destroyed ?f - fighter)
h) (behind ?f1 ?f2 - fighter)

2) Groundings: The PPDDL wargame grounding files are implemented scenarios of the wargame domain. The grounding files are made up of six sections: domain, objects, init, goal, goal-reward, and metric. The domain for these grounding files is the wargame domain. The objects for these files are equivalent numbers of fighters. For example, a 3 fighter wargame contains 6 objects of type fighter:

1) objects:
   a) b1 b2 b3 r1 r2 r3 - fighter

The init section describes the initial state of the scenarios. For example, a 3 fighter wargame may contain the following initial state:

1) init:
   a) (blue b1)
   b) (blue b2)
   c) (blue b3)
   d) (fifth-gen b1)
   e) (fourth-gen b2)
   f) (fourth-gen b3)
   g) (front b1)
   h) (behind b2 b1)
   i) (behind b3 b2)
   j) (last b3)
   k) (red r1)
   l) (red r2)
   m) (red r3)
   n) (fourth-gen r1)
   o) (fifth-gen r2)
   p) (fourth-gen r3)
   q) (front r1)
   r) (behind r2 r1)
   s) (behind r3 r2)
   t) (last r3)

The goal section specifies the goal state. For example, a 3 fighter wargame has the following goal state:

1) Objects:
   a) (and (destroyed r1) (destroyed r2) (destroyed r3) (or (not (destroyed b1)) (not (destroyed b2)) (not (destroyed b3))))

The goal state is reached by the entire red team being destroyed, and at least one blue team fighter not being destroyed. The goal-reward for the goal state is 1. The metric is to maximize the reward.

B. SSIPP

The source code for the SSIPP tool used to apply the LM-Cut heuristic as both a supervised teacher heuristic and as a comparison to the ASNet cost, win rate, and runtime is available to the public on GitHub here [4]. [5] provides a description of the implemented planner in these tools.

C. Action Schema Network

The source code for the ASNet framework used to develop and training these networks is available to the public on GitHub here [6]. This framework was developed by the authors of [1], and provides the framework for designing PPDDL domain and grounding files, as well as designing and training an ASNet.

1) Hyper-Parameters: The hyper-parameters used for training the networks in this work are equivalent for each trial and provided in tables I and II. Table I describes the network architecture used for each of the 5 trials in this work. The network architecture maintains 3 hidden layers, each with 16 units. Dropout of 0.25 is used in each layer to support regularization of network and prevent overfitting the training scenarios. The supervised teacher heuristic used to provide the truth labels given a state is LM-Cut. The network for each trial is trained for 10 epochs with minibatch sizes of 128 and a learning rate of 0.01. With a learning rate of 0.01 and 10 epochs; computational time-to-loss-convergence is decreased from the default learning rate of 0.0001 and 300 epochs.

2) Trials: A set of 5 trials are designed for this work. Each trial is broken down into particular training scenarios and testing scenarios to draw inferences on how networks trained on specific scenarios perform on other scenarios. Trial 1 involves training on 5 scenarios of 3 fighters per team, and testing on 15 scenarios of 3 fighters per team. Trial 2 involves training on 5 scenarios of 4 fighters per team, and testing on 15 scenarios of 4 fighters per team. Trials 1 and 2 provide intuition on how well networks perform when trained and tested on scenarios with equivalent numbers of fighters per team. Trial 3 involves training on 3 scenarios of 3 fighters per team and 3 scenarios of 4 fighters per team, and testing on 15 scenarios of 3 fighters per team and 15 scenarios of 4 fighters per team. Trial 3 provides intuition on how well a network performs when trained and tested on similar scenarios, while the scenarios do not have equivalent numbers of fighters. Trial 4 involves training on 5 scenarios of 4 fighters per team, and testing on 15 scenarios of 5 fighters per team. Trial 4 provides intuition on how well a network performs when trained on a set of scenarios with equivalent numbers of fighters per team, then tested on a set of scenarios with an greater, but equivalent, numbers of fighters. Trial 5 involves training on 3 scenarios of 3 fighters per team and 3 scenarios of 4 fighters per team, and testing on 9 scenarios, each with an increasing number of fighters per team from 5 to 13. Trial 5 provides intuition on how well a network performs when trained on scenarios with different numbers of fighters, and then tested on scenarios different, and higher, numbers of fighters.

For each scenario in each trial, the ASNet and the LM-Cut heuristic results are averaged over 30 runs, where each grounding has a different random seed determining the number of fourth and fifth generation aircraft on each team. Due to the statistical nature of this work, this can provide clearer insight into the results; rather than focusing on outliers.

TABLE I: Architectures

Architecture (Hidden Layers)	Dropout	Teacher Heuristic
16-16-16	0.25-0.25-0.25	LM-Cut

TABLE II: Hyper-Parameters

Epochs	Minibatch Size	Learning Rate
10	128	0.01

IV. Results

The results of the 5 Trials are presented in tables V and VI, and the runtime analysis of trial 5 is presented in table VII and in figures 1 and 2.

A. Results: Cost and Win Rate Analysis

Table V provides average results of the 5 trials when measured against the training scenarios. In all trials but trial 5, the number of wins out of 30 are higher for the LM-Cut Heuristic than for the ASNet. This lines up with the intuition that LM-Cut (because it is the teacher heuristic providing the truth labels to the ASNet) will have have a higher (or equal) number of wins statistically, than the ASNet. The cost of each trial is a measure of the average number of actions (of 30 runs) that were taken before either a dead-end state was reached or the goal state was reached. Note that both the runs in which the goal state was reached and those in which a dead-end state was reach were included in this average value. Interestingly, the average cost of the neural network runs were lower than the LM-Cut cost in each trial, except for trial 4. This could be for many reasons, but is possibly due to the fact that the ASNet did not learn the appropriate actions to take (the actions LM-Cut would have taken) in every state based on the training scenarios. However, since the win rate is greater than 50% in every trial, this indicates that it did learn some of the appropriate actions to take. Trial 4 is interesting because the win rate of the ASNet exceeded the winrate of the LM-Cut heuristic. However, the difference is small, and close to the win rate of the LM-Cut teacher heuristic of 21/30. The cost between the ASNet and the LM-Cut heuristic vary by only 0.15 indicating that it's likely that the ASNet for trial 4 correctly learned all of the actions to take given a state that the LM-Cut heuristic would have taken. However, it is equally as likely that this is a statistical anomaly due the probabilistic nature of the test runs, specifically because trials 2 and 4 had the same training set.

TABLE III: Trials: Training Scenarios

Trial	Number Of Fighters Per Team	Number Of Scenarios
1	3	5
2	4	5
3	Mixed 3 4	6 (3 of 3, 3 of 4)
4	Mixed 3 4	6 (3 of 3, 3 of 4)
5	Mixed 3 4	6 (3 of 3, 3 of 4)

TABLE IV: Trials: Test Scenarios

Trial	Number Of Fighters Per Team	Number Of Scenarios
1	3	15 of 3
2	4	15 of 4
3	Mixed 3 4	30 (15 of 3, 15 of 4)
4	Mixed 3 4	15 of 5
5	Mixed 3 4	9 (5 to 13)

TABLE V: Results: Training Scenarios

Trial	Ave. NN Cost	NN WR	Ave. LM-Cut Cost	LM-Cut WR
1	4.05	17.25/30	6.66	21.05/30
2	5.66	16.87/30	6.66	21.05/30
3	4.84	17.35/30	6.82	21.14/30
4	6.81	22.25/30	6.66	21.05/30
5	4.84	17.35/30	6.82	21.14/30

TABLE VI: Results: Test Scenarios

Trial	Ave. NN Cost	NN WR	Ave. LM-Cut Cost	LM-Cut WR
1	3.80	19.25/30	5.14	22.69/30
2	5.65	21.00/30	5.14	22.69/30
3	4.86	20.00/30	5.18	21.94/30
4	6.49	26.50/30	5.14	22.69/30
5	12.79	27.55/30	7.09	21.34/30

Table VI provides the results of the test data scenario runs for each trial. The results of the test trials are promising and indicate that the ASNets appropriately learned which actions to take in each of the trials; that is, the action that the LM-Cut heuristic would have taken. The ASNet win rates for trials 1, 2, and 3 are just slightly under win rates for the LM-Cut heuristic. The average costs are similar as well, with the exception of trial 1 which has a slightly lower cost and a slightly lower win rate. Trials 4 and 5 are the most interesting since in both trials the win rate is higher than the LM-Cut heuristic and close to 27/30. The exact reasoning for this is unknown; however, this could be due to the fact that the test scenarios for trials 4 and 5 contain higher numbers of fighters providing the ASNets with more opportunities to select the appropriate actions and terminate the red team fighters. In addition, the LM-Cut heuristic may have had to recalculate its position based on its state after each transition, where the neural network could contain hidden information about the ordering of the fighters that it was able to exploit. However; this remains speculation. These results indicate that ASNets generalize well to higher order problems.

B. Runtime Analysis

Figures 1 and 2 show two graphs of the runtime analysis of Trial 5 between the ASNet and LM-Cut Heuristic. Figure 1 shows a scatter plot of the runtime versus problem size of the actual data points collected during testing from table VII. As seen, both trend lines appear to be exponential. This is against what we would intuitively expect about the ASNet. Though the LM-Cut heuristic should scale exponentially as the problem size increases, intuitively, the ASNet should be more linear in nature; only increasing ever so slightly with the increase problem size. The reasoning for this is that ASNet neuron weights should be fixed, and require loading time plus a small amount of computation time to produce a recommended action in a particular state. Though we cannot clearly see a linear relationship here; this could be in part an issue related to the initial PPDDL grounding translation into enumerated predicates. It is not likely to continue along this exponential

TABLE VII: Results: Trial 5 Runtimes (s)

N	ASNet Runtime	ASNet Train + Runtime	LM-Cut Runtime
3	0.039667	5572.78819	0.000166
4	0.070292	5572.818516	0.000636
5	0.149784	5572.898007	0.003736
6	0.363373	5573.111597	0.018469
7	0.796418	5573.544641	0.074856
8	2.347297	5575.095521	0.324949
9	3.608478	5576.356701	0.473104
10	5.590633	5578.338857	1.127675
11	8.908236	5581.656701	2.165944
12	13.075251	5585.823474	4.714597
13	20.399599	5593.147823	7.871842

path as the problem size increases further, but to flatten out as N gets larger. Though the results on this small sample size, with the max grounding problem size being a scenario of 13 fighters per team, appears to be exponential, if we assume that this curve flattens as N increases, we can assume a point at which the ASNet will be computationally better than the LM-Cut Heuristic.

In order to illustrate this idea, figure 2 presents a plots of the ASNet training time + runtime vs. the LM-Cut runtime. In figure 2 the ASNet runtime is made up of the time spent training the ASNet plus the runtime of the ASNet for the problem size with the assumption of a linear trendline based off the scatter data from table VII, while the LM-Cut runtime is assumed to have an exponential trendline based off of the scatter data from table VII. This results in the ASNet's training time + runtime performance out pacing the LM-Cut heuristic's runtime performance as soon as the problem scenario reaches a size of more than 18 fighters per team. This indicates that for problem sizes greater than 18, it is better to train an ASNet on a set of smaller size problems and run it on the larger size problem, then to run the LM-Cut heuristic directly on the large size problem.

V. SOURCE CODE

The files developed for use for with the ASNet framework and SSIPP tool have been made publically available on GitHub here https://github.com/rpicard92/action-schema-networks under an MIT license. These files include the wargame PPDDL domain, PPDDL grounding files, grounding generator scripts, trial experiment files, the experiment results, and python scripts to collate the results of the trials.

VI. FUTURE WORK

It would be beneficial to utilize higher computational resources to test scenarios of higher orders against LM-Cut, and see if the ASNets runtime curve levels out into a more linear form as assumed. It would also be beneficial to do a guess and check hyper-parameters exploration when into comes to architectural and regularization parameters of these ASNets. Last, more trials of higher order problems would provide clearer insight into the ASNets internal representations of the problem, and whether the architecture can be exploited to capture the ordering of the fighter aircraft and provide a higher win rate than the LM-Cut teacher heuristic consistently such as in Trials 4 and 5.

VII. CONCLUSION

In this work we have presented a case study on a action schema networks applied to a stochastic shortest path wargaming problem domain. We have presented introductory information on the problem space as well the approach to training neural networks (ASNets) with an LM-Cut teacher heuristic to solve probabilistic wargamining problems. The work presented 5 test trials of various training and testing scenarios. The training scenarios were used to train ASNets on a set of wargaming scenario problems specified in by PPDDL grounding files based on a PPDDL wargaming domain. The ASNets were trained with truth data (best action to take give a state) provided by the LM-Cut planning heuristic. The testing scenarios (specified likewise) were tested on the trained ASNets and compared the LM-Cut heuristic results.

The results of the test trials 1, 2, 3 revealed that the ASNets were capable of learning which action the LM-Cut Heuristic would have taken, since the cost and win rates were similar the LM-Cut heuristic. The results of test trial 4 and 5 illustrated that it is possible for the ASNets to perform better than the LM-Cut heuristic when trained on smaller size problem scenarios and tested on larger size problem scenarios. This is possibly due to the ASNets storing an internal representation of the ordering of the fighter aircraft, which the LM-Cut heuristic does not directly account for.

The runtimes of the the ASNet and LM-Cut heuristic for trial 5 revealed both to be an exponential curve for problems of size 13 or less. It is more likely that the ASNet runtime will level out into a more linear curve as the problem size increases, since most of the time delay is likely related to the initiating the problem than solving for an action. The LM-Cut heuristic is expected to maintain an exponential growth rate as the problem size increases. Taking these assumptions into account, and extrapolating the results, it is revealed that with a scenario problem size of greater than 18 fighters per team, it is more time efficient to train an ASNet on smaller size problems and run it to solve the large size problem, than to solve the large size problem with the LM-Cut heuristic directly.

REFERENCES

[1] S. Toyer, F. Trevizan, S. Thiébaux, and L. Xie, "Action schema networks: Generalised policies with deep learning," in AAAI Conference on Artificial Intelligence (AAAI), 2018.

[2] B. Nebel and R. Mattmller, "Principles of ai planning." [Online]. Available: http://gki.informatik.uni-freiburg.de/teaching/ws1314/aip/aip12-handout.pdf

[3] M. Helmert and G. Roger, "Planning and optimization c20. landmarks: Cut landmarks lm-cut heuristic." [Online]. Available: https://ai.dmi.unibas.ch/_files/teaching/hs16/po/slides/po-c20.pdf

[4] M. M. V. Felipe W. Trevizan, "Ssipp." [Online]. Available: https://gitlab.com/qxcv/ssipp

[5] F. W. Trevizan and M. M. Veloso, "Depth-based short-sighted stochastic shortest path problems," Artif. Intell., vol. 216, no. 1, pp. 179–205, Nov. 2014. [Online]. Available: http://dx.doi.org/10.1016/j.artint.2014.07.001

[6] S. T. Sam Toyer1, Felipe Trevizan and L. Xie1, "Implementation of asnets." [Online]. Available: https://github.com/qxcv/asnets/tree/aaai18

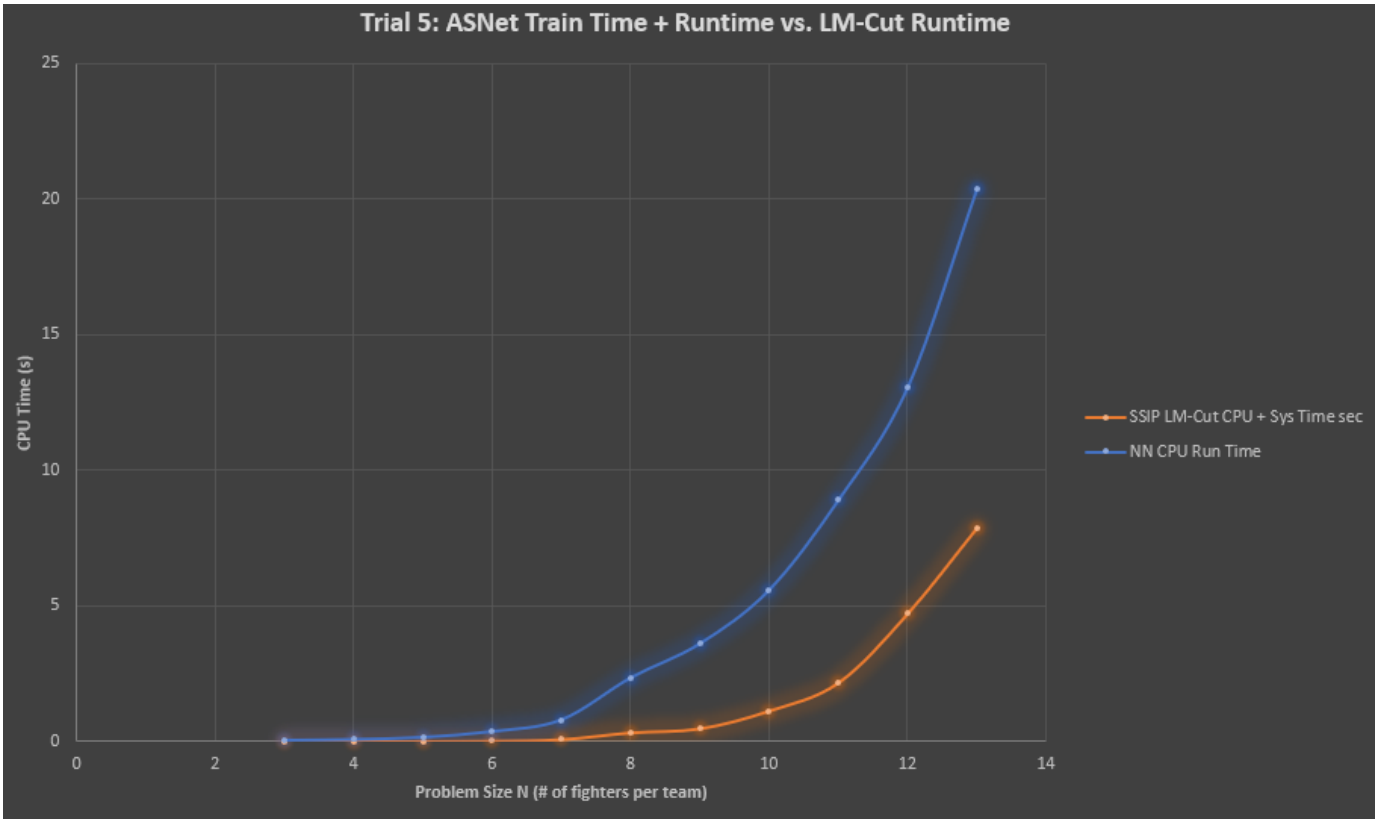

Fig. 1: Trial 5: Time vs. N

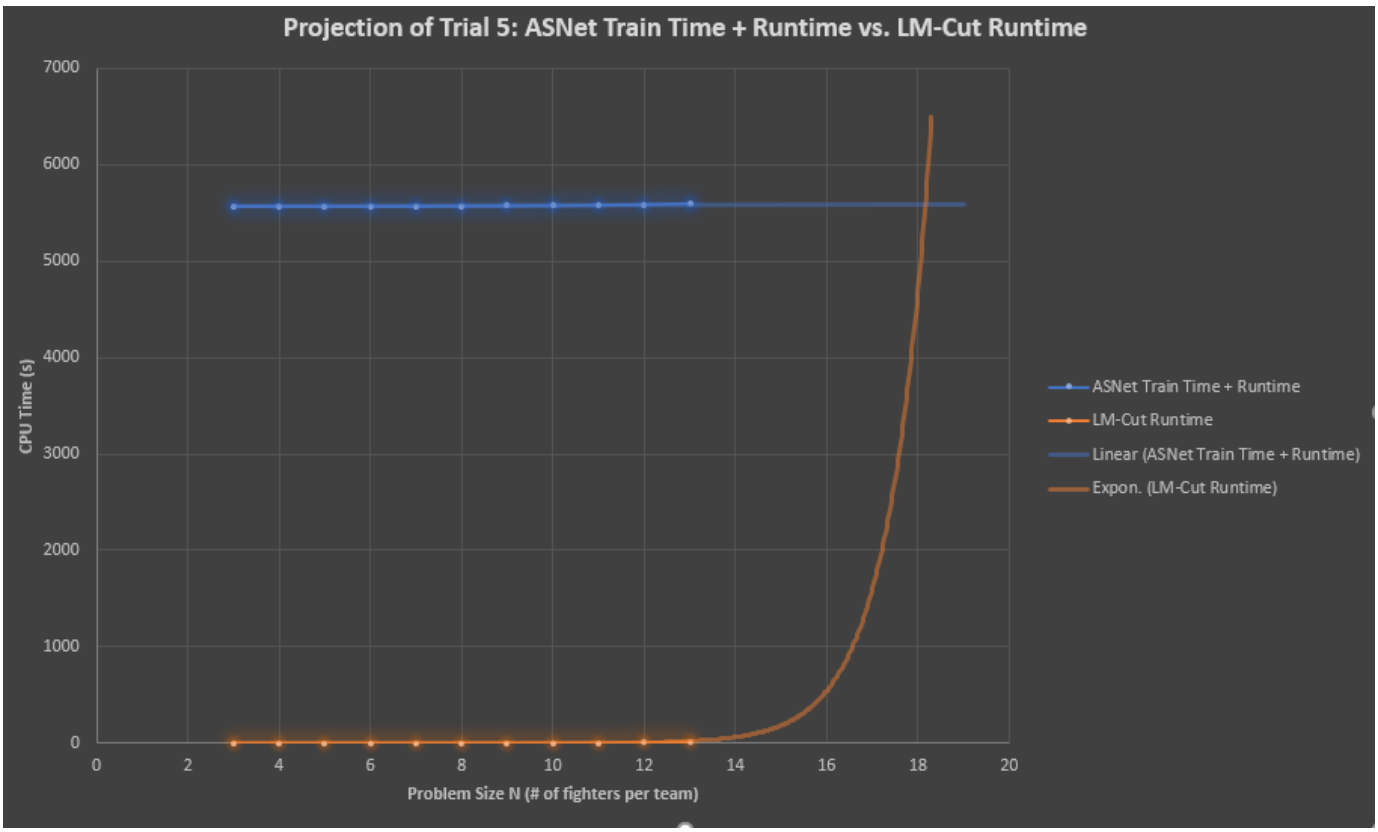

Fig. 2: Projected Trial 5: Time vs. N