1 MATHEMATICAL PROGRAM FORMALIZATION

The intuition of the formalization goes like this. We consider the action cost to handle an incident given some present state of knowledge to be the cost of the action(s) given that knowledge. The probes (also known as measurements) enhance our state of knowledge. We also consider increased costs of the incident as we delay action.

So given a set of measurements, the anticipated action cost is the weighted probability of the actions required based on each knowledge state times the probability that the measurements will put us in that knowledge state. We add to that action cost the cost of the measurements and the cost of the delay (which is usually positive) to handling the incident due to the time it takes to perform the measurements.

So, intuitively the total cost given a set M of measurements is the expected action cost given M and the action delay cost associated with taking the action after waiting for M + the cost of M + the cost of the extra delay, resulting from the incident itself, due to waiting for M to happen.

A state of knowledge or location region or just region for short is a set of values of location variables. Here, location is understood broadly to include locations and intensities, e.g. the location of the fire and the temperature; or the location of the field and the intensity of water deprivation.

Let's say our current state of knowledge is T. Let $f(e)$ be a probability distribution over location variables in T and $P(r|e, M)$ be the probability that the measurements will compute to a region of location values r from some non-empty set of measurements M given that the incident is in location e. Thus, different location values $e1$ and $e2$ may generate different and possibly overlapping regions. For example, a probe with detection range of 10 placed at location 50 will identify a region of [40..60] with probability 1 if the location of the object is at 43, but will identify a region of **not** [40..60] if the location of the object is 33. Similarly, more measurements may, through the intersection of confidence intervals, give smaller region specifications than fewer measurements.

Actions are things we do, such as sending in firetrucks or administering certain medicines. Cost of actions depends on the application: it could be money cost or it could be cost in lives or health outcomes. If we want to optimize several criteria (e.g. health cost first and then money), we look at outcomes that have the same values for the highest priority criterion and choose the one that does best for the next criterion. If actions conflict on the highest priority criterion (e.g. prescribing drug A and drug B if the two together are known to be lethal), then the cost of that set of actions is infinite.

For a region r, $\mathsf{MinCover}(r)$ is a minimal set of actions that covers r, i.e. that can take care of any value with region r. Finally $\mathsf{Cost}(\mathsf{MinCover}(r))$ is the cost of those actions (See Figure 1). So this is the expected action cost from the measurements M given current state of knowledge (region) T, denoted $actioncost(M, T)$ is:

$$\int_{e\in T} f(e) \int_{r\subseteq T} P(r|e, M) \times (\mathsf{Cost}(\mathsf{MinCover}(r)) + actiondelaycost(\mathsf{MinCover}(r), time(M))) \tag{1}$$

Thus the total cost is

$$min_M(actioncost(M, T) + cost(M) + delaycost(M)) \tag{2}$$

Let's look at this intuitively. Suppose the measurements are very cheap, accurate, and fast to do and $\mathsf{Cost}(\mathsf{MinCover}(T))$ is high. The reason may be that there are a few values in T that call for very expensive actions but are unlikely to occur. (Sometimes, $\mathsf{Cost}(\mathsf{MinCover}(T))$ may be infinite if there are locations $e1$ and $e2$ within T that call for conflicting actions).In such cases, it's worthwhile to perform the measurements M because M may allow us to use far cheaper actions and the delay cost will be low. Conversely, if the measurements are slow, inaccurate, and

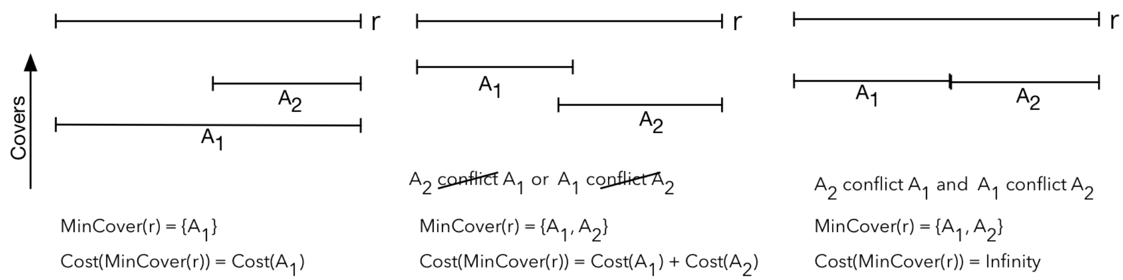

Fig. 1. Visual Interpretation of MinCover(r) and Cost(MinCover(r)) given disjoint or (partially) covering actions and conflicts. Note that the covering as well as the conflict relationship are region specific. It is possible for an action to cover another for region r but not for region r'. Similarly it is possible for two actions to conflict on region r but not r'.

expensive or the likelihood that the actioncost will be reduced substantially is low, then we should set M to null and pay Cost(MinCover(T)).

If no set of measurements eliminates conflicting actions, then the algorithm reports the probability of success of each non-conflicting subset of actions.

Here are some concrete examples.

In a medical setting, we might have narrowed down our diagnosis to several possible diseases. If the different diseases entail different actions , some of which may conflict, then we may want to consider various blood measurements, x-rays, etc. Those would be the measurements. If the patient is very sick and treatment is critical, we may decide to go ahead and act even in the absence of the measurements. On the other hand, if the patient is stable, we may decide it's better to take the measurements and then to embark on what we hope will be a minimal action. The higher the likelihood that the measurements will give a good differential diagnosis, the more valuable are the measurements. Our actions ultimately depend on the measurements. The diagnoses are shorthands that are useful for people and a rule-based system may use them (again as shorthands) to determine treatment actions.

In a drought setting where we are deciding whether to irrigate some fields, irrigation alternatives are the actions. Measurements include soil measurements, rain measurements and potentially weather forecasts. Deploying measurements may mean deploying sensors to local fields in case there are many micro-climates (as in a mountainous region) in which case the actions we take for different fields may be different. If we don't deploy, measurements may be free, but the consequences of the drought could become worse.

In a fire setting, measurements are determinations of the size and heat of a fire, the types of flammable matter near the fire, and the number of people at risk. Actions are to send firefighters to the scene. Waiting could make the fire worse or could allow the fire to die out on its own. Also, if we decide to wait for measurements and the fire does get worse, then the action cost associated with sending the firefighters also increases.

In the submarine hunting setting, measurements are the probes sent to detect the position of the submarine and actions are red alert or yellow alert.

1.1 NP-Completeness of the Problem: motivating heuristic solutions

Finding the minimum set of measurements is an NP-complete problem as can be seen by the following simple argument: Suppose that there is a set of regions S and each can be handled by a single expensive action. There is also a set of measurements M each of which excludes one or more regions. What is the minimum subset of measurements M that would eliminate all regions? This is a hitting set problem.

A completely analogous argument shows that finding the set of actions having minimum cost to cover a region r is also NP-complete: There is a set of regions S that we need to take care of. Each action covers some subset of S. Let's say all the actions cost the same. Then it's a hitting set problem to decide on the smallest set of actions that cover all of S.

For this reason, we use heuristic techniques like genetic algorithms both to find $actioncost(M, T)$ and to minimize M. However, there are ways to eliminate certain measurements from consideration. We start with simple ones.

A measurement is *useless* if it doesn't affect any of the variables that determine actions (e.g. measuring a flood level during a fire). Those can be discarded immediately.

A measurement m with respect to a variable x and region Q is characterized by a function $conf(m, x, v)$ for all values v lying in Q defined as follows: if the actual value of x is v, then the confidence interval $conf(m, x, v)$ includes v. Measurement m is more accurate than m' with respect to Q with respect to x means that for all values v lying in Q, $conf(m, x, v)$ is contained in $conf(m', x, v)$.

A measurement m *universally dominates* measurement m' if m is no more expensive than m' for any location variable, m is at least as accurate as m' and for at least one location variable m is more accurate. In such a case, we can discard m'. However, we can also discard m' even if m *conditionally dominates* m' with respect to some region Q which would mean that m is no more expensive than m' for any location variable whose value lies within Q, m is at least as accurate as m' and for at least one location variable whose value lies within Q, m is more accurate. For example, if dogs are better at sniffing out drugs than people and cost less, then use dogs for that purpose.

In some situations we can decompose the problem using dynamic programming. For example, if each measurement pertains to a different variable, then we can treat the measurements of each variable separately. We need to think of other mechanisms like this.

1.2 Confidence Arithmetic

Adding/Subtracting and Multiplying uncertain values is based on the website http://web.uvic.ca/ jalexndr/192UncertRules.pdf

Fusion functions

Fusing x +/- d1 and y +/- d2 is the same as fusing [x-d1 .. x+d1] and [y-d2 .. y+d2]. Fusing that gives us [max(x-d1, y-d2) .. min(x+d1, y+d2)]. If this result is improper (i.e. if max(x-d1, y-d2) > min(x+d1, y+d2), then the confidence interval is empty (we know nothing).

Sometimes, we need disjoint intervals. For example if the probe at 65+epsilon does not detect the submarine then the submarine is at [0..45] or (85+epsilon .. 100] if the entire interval is 100 kilometers long.

So we need to intersect disjoint intervals and to handle negative intervals. To intersect a disjoint interval with a single interval, intersect each arm separately and maintain the disjunction. So, intersecting [a..b] or [c..d] with [e..f] yields intersect([a..b], [e..f]) or intersect([c..d], [e..f]). To intersect two disjoint intervals, there will be potentially four arms. So, to intersect [a..b] or [c..d] with [e..f] or [g..h], we get intersect([a..b], [e..f]) or intersect([a..b], [g..h]) or intersect([c..d], [e..f]) or intersect([c..d], [g..h]).

We also need to intersect intervals with negations of intervals. intersect([a..b], not [c..d]) = [a..b] if d < a or b < c. Otherwise (a <= d) and (b >= c), [a..c) or (d..b]

So applying this to our warm-up puzzle, intersect([40..60],not (45..85]) = [40..45] or [85..60] = [40..45], since [85..60] has no information

There can also be a degradation function that is a function of time. So if the confidence interval is [x-d1 .. x+d1] when a measurement is made, the degradation function may widen out the confidence interval based on [x-d1-degrade(deltat, x, d1) .. x + d1 + degrade(deltat, x, d1)], where deltat is the time since the measurement.

degrade(deltat, x, d1) might be something like f*deltat where f is some factor that says how much the uncertainty increases per second. And should be strictly positive. The degradation function is a function of the variable being measured.