Dynamic On-the-fly Generation of Scan Schedules

Ants Final Demo
November 2001

Bruno Dutertre
System Design Laboratory
SRI International
e-mail: bruno@sdl.sri.com

Outline

Single Platform DSS

- Problem Statement
- Construction of Regular Schedules
- Hardness and Phase Transitions
- Finding Optimal Δ's
- Implementation

Multiplatform DSS

Conclusion

- Known Limits and Possible Improvements
- Generalizations

Problem Statement

Initial Specifications

Input

- n: number of frequency bands
- Emitter table
- For each emitter type E: a weight W_E and minimal coverage $p_E \in (0, 1]$

Objective

- Compute in real time a scan schedule S that maximizes

$$F = \sum_{E} W_E P_S(E)$$

where $P_S(E)$ is the probability of detecting E with S.

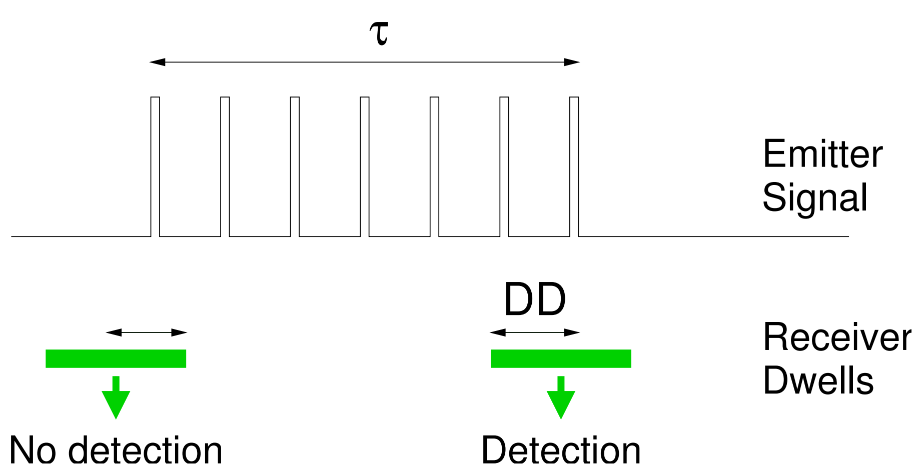
Coverage Constraints

- Make sure that $P_S(E) \geqslant p_E$ for all E.

Emitter Characteristics

For each emitter type E

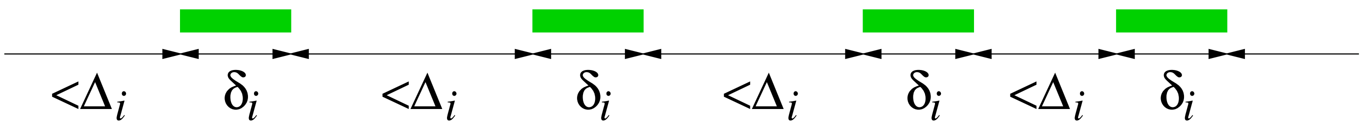
- i: frequency band
- τ_E: nominal illumination time
- DD_E: duration to detect



Central Concept: Regular Schedules

Definition

- Characterized by parameters $\delta_1, \ldots, \delta_n$ and $\Delta_1, \ldots, \Delta_n$.
- All dwells for band i are of length δ_i.
- Two successive dwells for band i are separated by a delay no more than Δ_i.

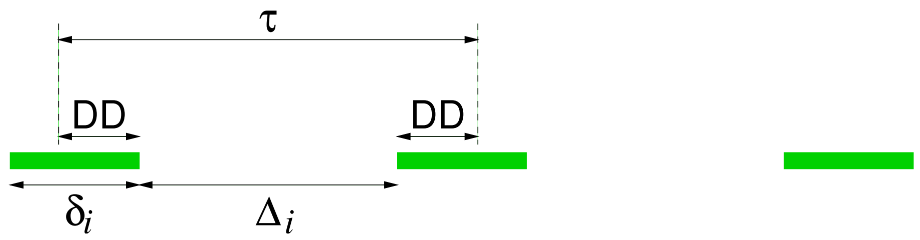


Important property

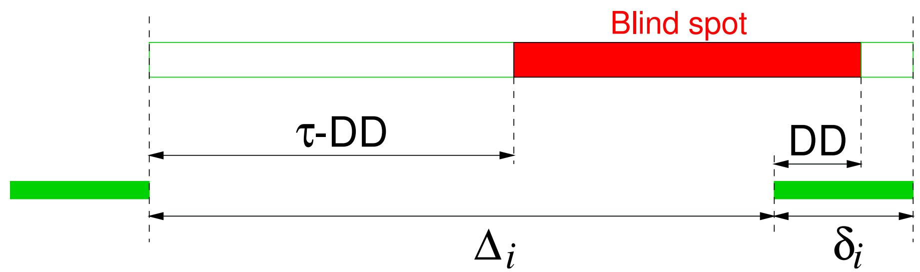
- If E is an emitter of band i, then a lower bound on $P_S(E)$ can be easily obtained from δ_i and Δ_i.

Detection Probabilities

$$\delta_i \geqslant DD_E \text{ and } \Delta_i \leqslant \tau_E - 2DD_E \quad \Rightarrow \quad P_S(E) = 1$$



$$\delta_i \geqslant DD_E \text{ and } \Delta_i > \tau_E - 2DD_E \quad \Rightarrow \quad P_S(E) \geqslant \frac{\delta_i + \tau_E - 2DD_E}{\delta_i + \Delta_i}$$



Reformulating the Problem

$$\text{Let } Q(E) \;=\; \min(1, \frac{\delta_i + \tau_E - 2DD_E}{\delta_i + \Delta_i})$$

New Objective Function

$$G(\Delta_1, \ldots, \Delta_n) \;=\; \sum_E W_E\, Q(E)$$

Bounds on Δ_i

- Upper bound: The constraints $Q(E) \;\geqslant\; p_E$ for emitters in band i give

$$\Delta_i \;\leqslant\; B_i$$

  where B_i depends on δ_i and on the emitters in band i.

- Lower bound: There is A_i below which $Q(E) = 1$ for all emitters in band i.

Reformulating the Problem (cont'd)

Dwell times are fixed: δ_i is the maximal DD_E among emitters E in band i.

New optimization problem:

- Objective:

$$\text{maximize} \;\; G(\Delta_1, \ldots, \Delta_n)$$

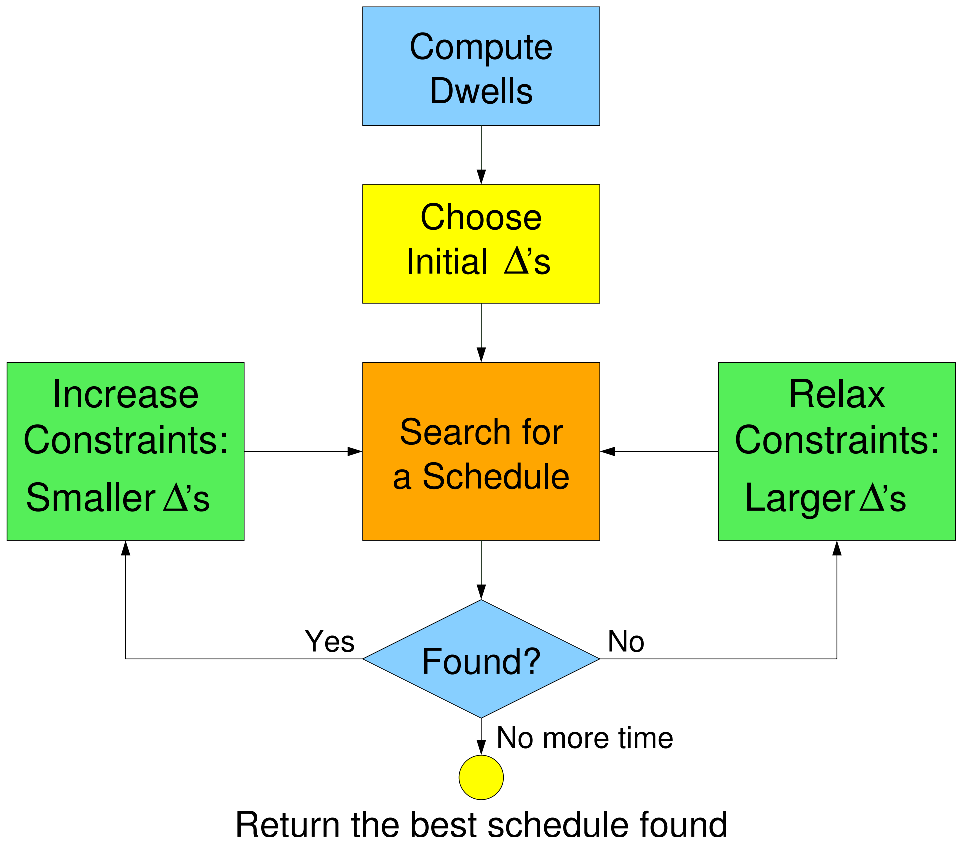
- Coverage constraints:

$$\begin{array}{ccccc} A_1 & \leqslant & \Delta_1 & \leqslant & B_1 \\ & & \vdots & & \\ A_n & \leqslant & \Delta_n & \leqslant & B_n \end{array}$$

- Feasibility constraint:

  Make sure that there exists a regular schedule S for $\delta_1, \ldots, \delta_n$ and $\Delta_1, \ldots, \Delta_n$.

Overview of the DSS Approach



Schedule Construction Algorithm

Construction of Regular Schedules

Objective

- Given $\delta_1, \ldots, \delta_n$ and $\Delta_1, \ldots, \Delta_n$, compute a regular schedule S for these parameter or determine that no such schedule exists.

Complexity

- This is an NP-hard problem

Necessary Conditions for Feasibility

- We must have $\delta_i \leqslant \Delta_j$ for $i \neq j$ and

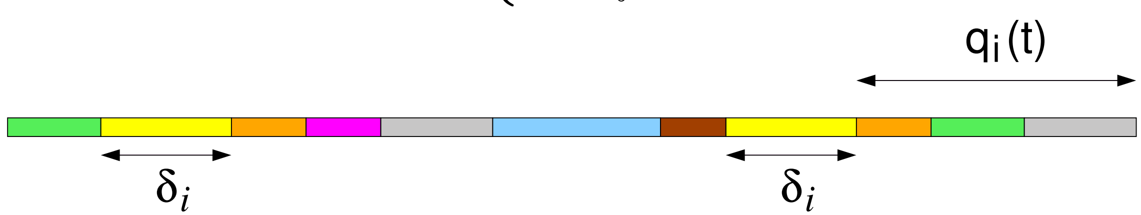
$$\sum_{i=1}^{n} \frac{\delta_i}{\delta_i + \Delta_i} \leqslant 1$$

Translation to a Graph Problem

Regular Schedule:

- S is an infinite sequence of band indices $(f_t)_{t \in \mathbb{N}}$
- At step t, let $q_i(t)$ be the delay since the last occurrence of band i:

$$q_i(0) = 0$$
$$q_i(t+1) = \begin{cases} q_i(t) + \delta_{f_t} & \text{if } f_t \neq i \\ 0 & \text{if } f_t = i. \end{cases}$$



Since S is regular, we have

$$\forall t \in \mathbb{N} : q_i(t) \leqslant \Delta_i$$

Translation to a Graph Problem (cont'd)

Directed graph defined by the δ's and Δ's

- Vertices:

  Tuples $q = (q_1, \ldots, q_n)$ such that $q_i \leqslant \Delta_i$ for $i = 1, \ldots, n$.
- Edges:

  $q \longrightarrow q'$ if there is j such that

$$\begin{aligned} q'_j &= 0 \\ q'_i &= q_i + \delta_j \text{ if } i \neq j \end{aligned}$$

Properties

- A regular schedule S is an infinite path in this graph.
- Since the set of vertices is finite, there is a regular schedule if and only if the graph has a circuit.

Algorithm

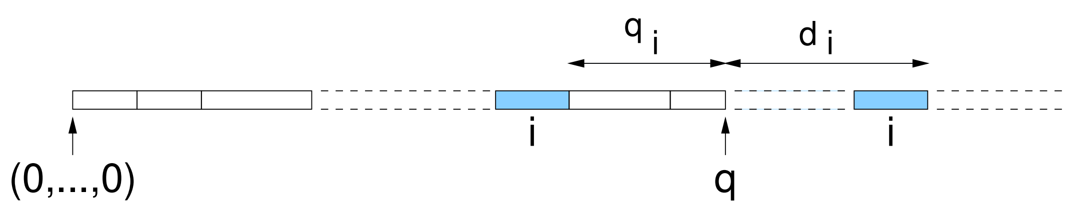
Naïve Algorithm

- Search for a circuit in the graph using depth-first search starting from $(0, \ldots, 0)$.
- (The graph is too large to compute transitive closure or use other algorithms that are polynomial in the graph size)

Optimizations

- Pruning to detect dead-ends early.
- Subsumption: finding a circuit is not necessary, a weaker property is sufficient.
- Heuristics to order the search.

Pruning Technique



- q: last state on the current path, during depth-first search
- $d_i = \Delta_i + \delta_i - q_i$: deadline for the next dwell in band i
- if q is on an infinite path, we can add a sequence of n dwells after q, one for each band, without missing any deadline
- Property: whether such a sequence of n dwells exits can be efficiently checked via a test based on earliest-deadline-first scheduling (EDF)
- Pruning: if this EDF test fails, q is *not* on an infinite path: no need to explore further
- Generalization: consider more than one dwell per band

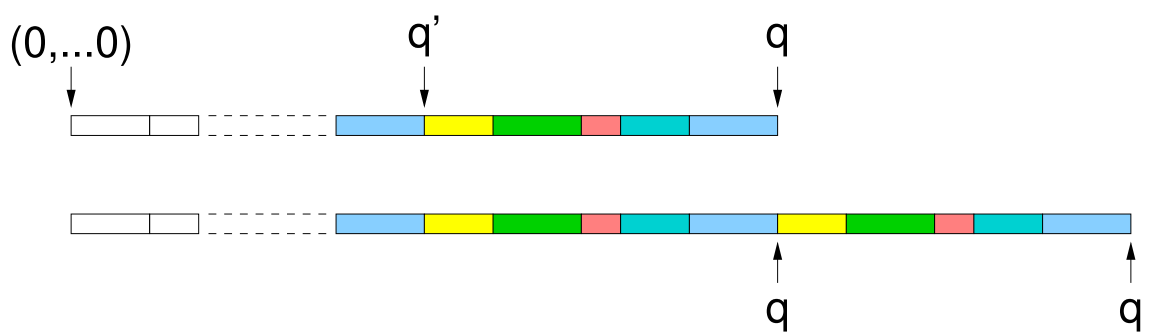
Subsumption

Definition

- q subsumes q' if $q \leqslant q'$, that is, $q_i \leqslant q'_i$ for $i = 1, \ldots, n$.
- if $q \leqslant q'$ then all sequences of bands admissible in q' are admissible in q

Consequence

- instead of searching for a circuit, we can stop exploration whenever we reach a q that subsumes a preceding state q'



Hardness Estimation
Phase Transitions

Estimating Hardness

Objective

- An instance consists of $2n$ parameters $\delta_1, \ldots, \delta_n$ and $\Delta_1, \ldots, \Delta_n$.
- We need to determine a priori whether an instance is likely to be feasible of not.
- This is essential to achieve "good enough/soon enough" guarantees.

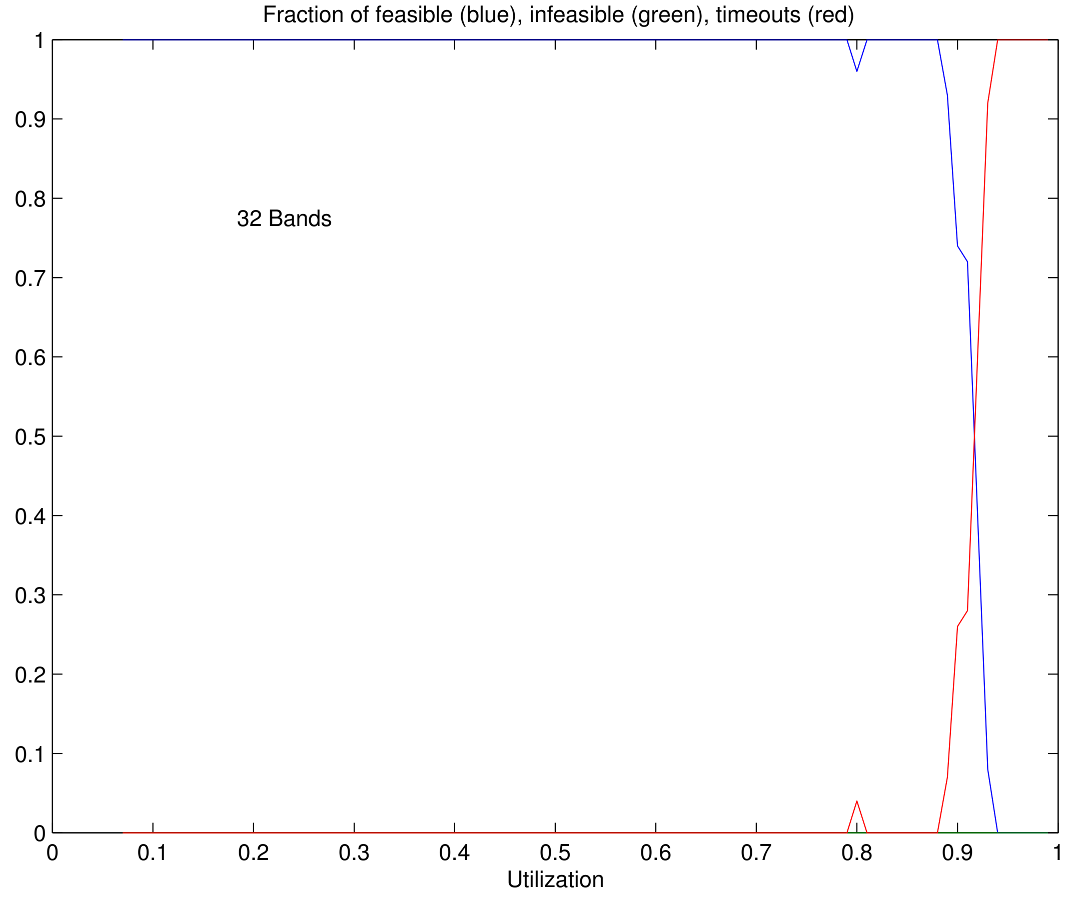
Experiments on Random Instances

- Show that the utilization can be used to predict hardness

$$U = \sum_{i=1}^{n} \frac{\delta_i}{\delta_i + \Delta_i}$$

- On randomly generated instances, with δ_i and Δ_i uniformly distributed, we see a phase transition around $U = 0.9$, independent of n
  - Almost all instances with $U < 0.8$ are feasible
  - Almost no instance with $U > 0.9$ is feasible

Phase Transition on Random Instances



More Realistic Experiments

DSS Context

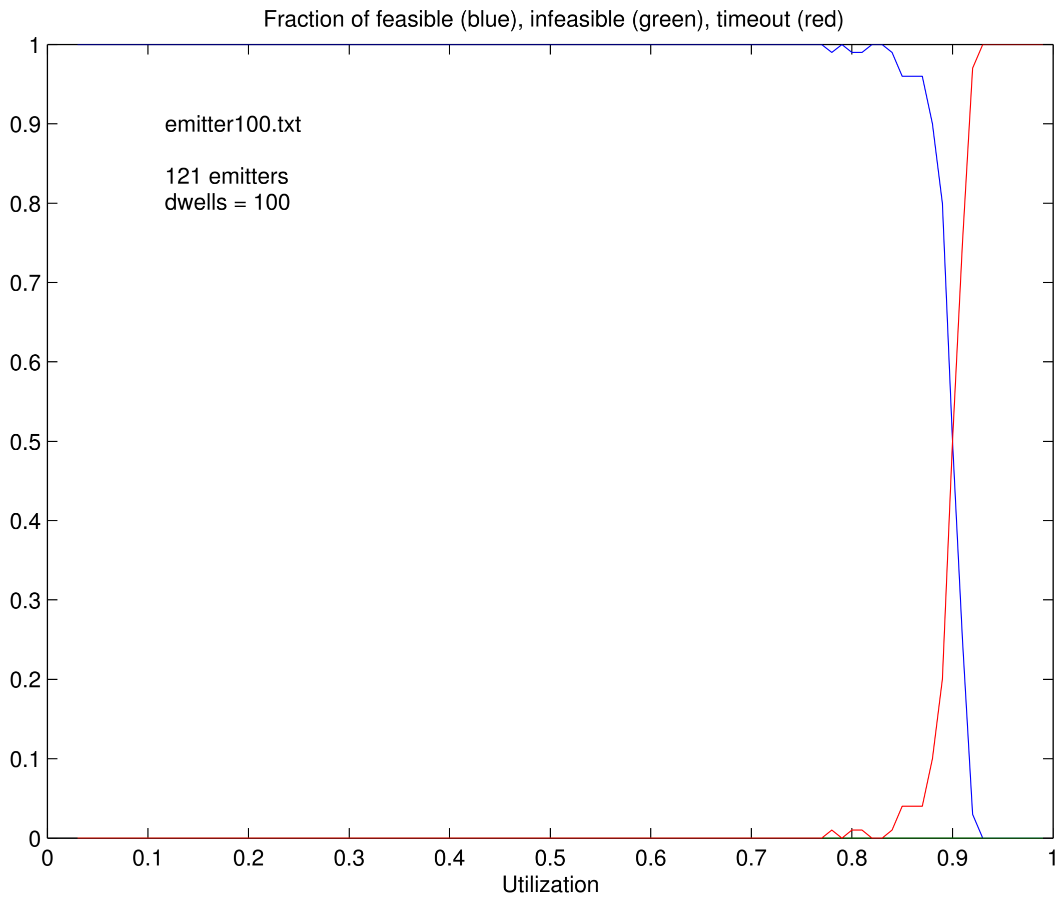
- Only $\Delta_1, \ldots, \Delta_n$ vary.
- The dwells $\delta_1, \ldots, \delta_n$ are fixed a priori by the emitter table.
- Bounds on $\Delta_1, \ldots, \Delta_n$ are also given a priori:

$$A_i \leqslant \Delta_i \leqslant B_i.$$

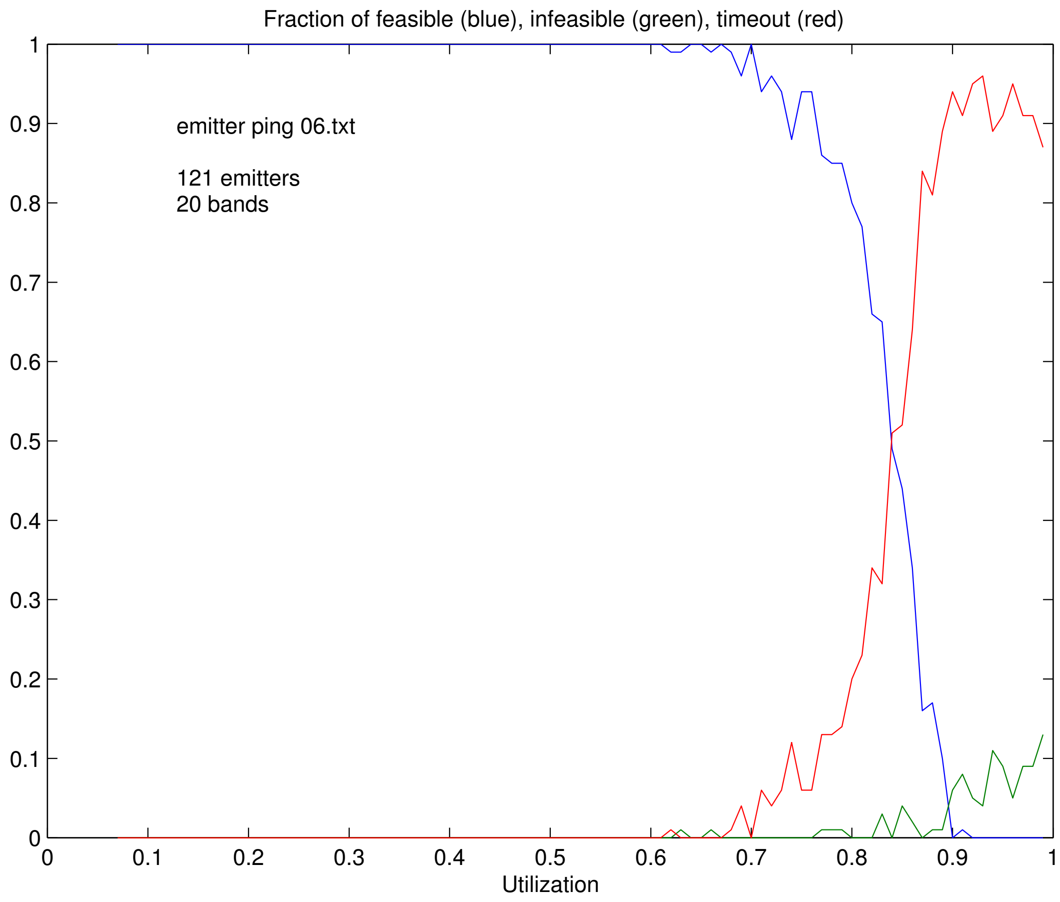
Experiments

- For a fixed emitter table, generate random instances with Δ_i uniformly distributed between A_i and B_i
- Record how the number of feasible/infeasible instances varies with U

Emitter Table 1



Emitter Table 2



Empirical Results

Results

- We still observe a phase transition for all the emitter tables we tried
- The sharpness and location of the transition vary with the emitter table
- Large dwells and small A_i's cause the transition to move left (more instances are hard)

Conclusion

- Experiments validate the use of U as a hardness indicator.
- Variation of hardness with U must be assessed for each emitter table and coverage parameters.
- Two utilization bounds can be estimated empirically
  - U_{easy}: below which all instances are feasible
  - U_{hard}: above which (almost) no instance is feasible

Computing $\triangle$'s

Computing Δ's

Objectives

- Find *feasible* $\Delta_1, \ldots, \Delta_n$ that maximize the function

$$G = \sum_E W_E Q(E)$$

and satisfy the constraints

$$A_i \leqslant \Delta_i \leqslant B_i.$$

Known Properties

- Determining feasibility is NP-hard
- Feasibility is related to

$$U = \sum_{i=1}^{n} \frac{\delta_i}{\delta_i + \Delta_i}$$

  - $U \leqslant U_{\text{easy}}$: very likely to be feasible
  - $U \geqslant U_{\text{hard}}$: very unlikely to be feasible

Computing Δ's (cont'd)

The DSS problem can be decomposed in two steps:

Optimization Problem

- For an utilization bound U_0, find $\Delta_1, \ldots, \Delta_n$ that maximize

$$G = \sum_E W_E Q(E),$$

under the constraints

$$A_i \leqslant \Delta_i \leqslant B_i,$$

$$\sum_{i=1}^{n} \frac{\delta_i}{\delta_i + \Delta_i} \leqslant U_0.$$

Schedule Construction

- For the optimal solution $\Delta_1, \ldots, \Delta_n$ to this problem, try to construct a regular schedule S.

Selecting U_0

Needs

- The two previous steps are iterated for several values of U_0
- If step 2 succeeds, a solution is found: U_0 is increased to attempt to find a better schedule
- If step 2 fails, we need to relax the constraints by reducing U_0

Approach Implemented

- Use dichotomy: maintain an interval $[U_1, U_2]$ and take $U_0 = \frac{U_1+U_2}{2}$
- Initially, $U_1 = U_{\text{easy}}$ and $U_2 = U_{\text{hard}}$
- If step 2 succeeds, set $U_1 = U_0$ otherwise set $U_2 = U_0$
- Iterate until $U_2 - U_1$ is small enough

Optimal strategy could be computed by solving a Markov Decision Process

Solving the Optimization Problem

New Variables

$$x_i = \frac{1}{\delta_i + \Delta_i}$$

then for any E in band i,

$$Q(E) = \min(1, (\tau_E + \delta_i - 2DD_E)x_i) = \min(1, \alpha_E x_i)$$

The problem is now almost linear:

Maximize

$$G = \sum_E W_E \min(1, \alpha_E x_i)$$

under the constraints

$$\frac{1}{B_i + \delta_i} \leqslant x_i \leqslant \frac{1}{A_i + \delta_i}$$

$$\sum_{i=1}^{n} \delta_i x_i \leqslant U_0$$

Idea Behind the Algorithm

Aggregate weights

- Take all emitters in band i and sort them in increasing order of $1/\alpha_E$, say

$$1/\alpha_{E_1} \leqslant 1/\alpha_{E_2} \leqslant \ldots \leqslant 1/\alpha_{E_m}.$$

- Compute the aggregate weights:

$$\mathsf{agw}(E_j) \;=\; \sum_{k=j}^{m} W_{E_k}$$

- Properties
  - If $1/\alpha_{E_{j-1}} \leqslant x_i$ and $x_i + \varepsilon \leqslant 1/\alpha_{E_j}$ then increasing x_i by ε increases G by $\varepsilon \times \mathsf{agw}(E_j)$
  - Increasing x_i by ε has a cost of $\varepsilon \times \delta_i$ in terms of utilization.

Algorithm Overview

Optimal solution is found as follows:

- Sort all the emitters in decreasing order of the ratios $\mathsf{agw}(E)/\delta_j$:

$$\mathsf{agw}(E_1)/\delta_{i_1} \geqslant \mathsf{agw}(E_2)/\delta_{i_2} \geqslant \ldots \geqslant \mathsf{agw}(E_N)/\delta_{i_N}$$

- Iteratively compute $x_1, \ldots, x_n$ so as to ensure

$$Q(E_1) = 1, \ldots, Q(E_M) = 1$$

  for as many of these emitters as possible (i.e., for the largest possible M).

- Stop when $\sum_{i=1}^{n} \delta_i x_i = U_0$.

Implementation

Implementation

Software

- Around 10K lines of C
- Linux compatible
- Includes DSS algorithm plus all algorithms for generating and solving random instances
- Output of DSS is a scan schedule given as a CDW table

Performance

- DSS computes a schedule in 1-2 seconds, depending on parameter settings, on a 400MHz Pentium III
- Experimental evaluation performed by BAE System

Multiplatform DSS

Overview

Rely on fast schedule construction

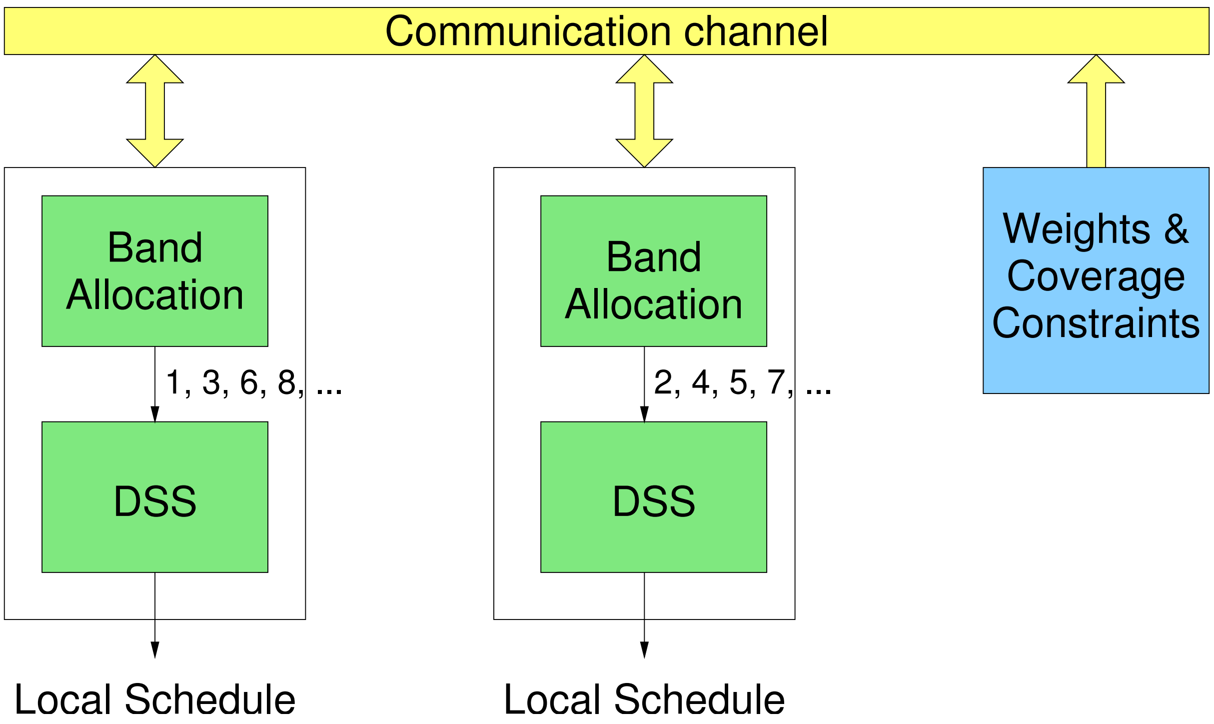
Distribute load accross the platforms

- Each platform is assigned a subset of the bands to focus on
- The weight of emitters in other bands is set to 0
- This gives a local scan-scheduling problem that is solved by DSS

Experimental Evaluation

- Based on a simplified emitter simulator that generates on/off files for each platform
- Can also use on/off files from BAE

Architecture



Band Allocation Approach

Objective

- Ensure that all bands are covered (allocated to at least one platform)
- Balance the load among the platforms

Approach

- All platforms receive the same weight and coverage data, and perform the same computation
- They all compute the total weight of each band:

$$W_i = \sum_{E \in \mathsf{Band}_i} W_E$$

- Bands are then sorted in decreasing order of W_i/δ_i.
- Bands are partitioned into N balanced subsets (currently $N = 2$) using a heuristic similar to the bin packing best-fit heuristic
- Platform j is assigned partition j.

Conclusion

Main Outcomes of the Work

Feasibility and benefits of on-line DSS generation

- Computing DSS in real time is possible even though the problem is NP-hard in general
- Simulation shows improved detection performance of adapting scan-schedule to changing mission priorities
- Online schedule construction algorithm enables dynamic cooperation between multiple platforms

Main Innovation

- Combination of graph exploration algorithms with hardness prediction based on utilization.

Limits and Possible Extensions

Limits of Regular Schedules

- Too restrictive in some cases
- A regular schedule requires $\delta_j \leqslant \Delta_i$ whenever $i \neq j$. This rules out certain emitter tables.
- In a regular schedule δ_i is the maximal DD_E among emitters in band E. This may be expensive if high-weight emitters in band i have DD_E much smaller than the maximum.

Possible Solutions

- Use non-regular schedules where dwells in a band have different lengths
- Graph exploration technique generalizes to this type of schedules without much problem.
- Generalization of hardness estimation techniques less clear.

Limits and Possible Extensions (cont’d)

Coverage Constraints

- Useful in the multiplatform case for robustness: no emitter is totally ignored by a platform
- But this can limit how well DSS does in overconstrained cases

Alternative: No Coverage Requirements

- Allows some emitter types to be ignored completely
- Should allow DSS to work beyond the “257 limit”
- Straightforward to implement, but has a nontrivial impact on the hardness prediction

Other Applications

- Graph algorithms and heuristics could be applicable to many other types of scheduling problems
- Examples
  - Task scheduling in RTOS
  - Bus scheduling in TTA or similar architectures
  - Communication scheduling in wireless networks

The relation between utilization and hardness should also generalize to these examples