Coordination of Robotic Networks: On Task Allocation and Vehicle Routing

Francesco Bullo

Center for Control, Dynamical Systems & Computation
University of California at Santa Barbara
http://motion.mee.ucsb.edu

Electrical and Computer Engineering Department
Carnegie Mellon University
April 30, 2009

Acknowledgements

Stephen L. Smith

Shaunak D. Bopardikar

SLS, FB: "Monotonic target assignment for robotic networks," *IEEE Trans Automatic Ctrl*, 54 (10), 2009
SLS, SDB, FB: "Finite-time pursuit of translating targets in a dynamic and stochastic environment," *Proc CDC*, Shanghai, 2009, submitted

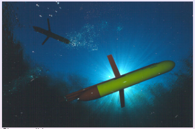
SDB, SLS, FB, JH: "Dynamic vehicle routing for translating demands," *IEEE Trans Automatic Ctrl*, 2009, submitted

Applications of autonomous systems

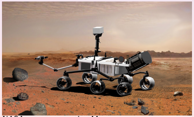
- Unmanned vehicles
- Equipped with suite of sensors
- Inaccessible environments

Civilian applications:

- Environmental monitoring:
  - Measure weather systems
  - Observe animal species
  - Detect and assess wildfires
- Search and rescue missions
- Space exploration
- Monitoring infrastructure



Slocum glider



NASA – next generation Mars rover

Applications of autonomous systems

Military applications:

- Surveillance
- Reconnaissance missions
- Perimeter defense and security
- Expenditures of \$60 billion over next 10 years

Globalhawk

Aerovironment Wasp

The future of autonomy

Current missions (typical scenario):

- single vehicle or few decoupled vehicles
- pre-specified task
- tightly coupled with human control

Future missions

1. Fleets (swarms) of networked vehicles
2. Complex sets of tasks that evolve during execution
3. Increased autonomy, humans as supervisors

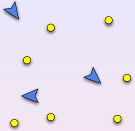
Requires real-time task allocation and vehicle routing

Task allocation

Given:

- a group of vehicles, and
- a set of tasks

Task example:
take a picture at a location



Task allocation

Decide which vehicles should perform which tasks.

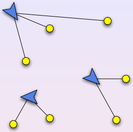
- Centralized: operator assigns vehicles to tasks
  (requires vehicle positions, workloads, etc.)
- Distributed: vehicles divide tasks among themselves

Task allocation

Given:

- a group of vehicles, and
- a set of tasks

Task example:
take a picture at a location



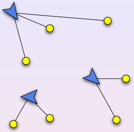
Task allocation

Decide which vehicles should perform which tasks.

- Centralized: operator assigns vehicles to tasks
  (requires vehicle positions, workloads, etc.)
- Distributed: vehicles divide tasks among themselves

Vehicle routing

Given:
An allocation of tasks to vehicles

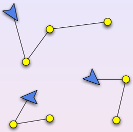


Vehicle routing

Determine a path that allows each vehicle to complete its tasks.

- Task A is of higher priority than task B
- A task requires multiple vehicles: vehicles need to rendezvous
- Task locations are not stationary

Vehicle routing



Given:

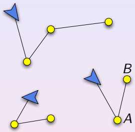
An allocation of tasks to vehicles

Vehicle routing

Determine a path that allows each vehicle to complete its tasks.

- Task A is of higher priority than task B
- A task requires multiple vehicles: vehicles need to rendezvous
- Task locations are not stationary

Vehicle routing



Given:

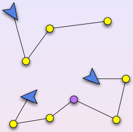
An allocation of tasks to vehicles

Vehicle routing

Determine a path that allows each vehicle to complete its tasks.

- Task A is of higher priority than task B
- A task requires multiple vehicles: vehicles need to rendezvous
- Task locations are not stationary

Vehicle routing



Given:

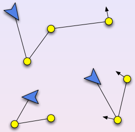
An allocation of tasks to vehicles

Vehicle routing

Determine a path that allows each vehicle to complete its tasks.

- Task A is of higher priority than task B
- A task requires multiple vehicles: vehicles need to rendezvous
- Task locations are not stationary

Vehicle routing



Given:

An allocation of tasks to vehicles

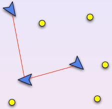
Vehicle routing

Determine a path that allows each vehicle to complete its tasks.

- Task A is of higher priority than task B
- A task requires multiple vehicles: vehicles need to rendezvous
- Task locations are not stationary

Dynamic and distributed aspects

Distributed:
Vehicles have only local information



Dynamic:

- Existing tasks evolve over time
- New tasks arise in real-time
- Number of vehicles changes

Complete solution cannot be computed off-line.

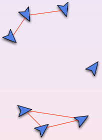
As new information becomes available, vehicles must

- re-allocate tasks
- re-plan paths

Technical approach: structure, fundamental limits, efficient algorithms

For a distributed/dynamic problem:

1. Identify underlying problem structure
   e.g., adimensional analysis, intrinsic regimes,
   phase transitions in parameter space
2. Determine fundamental limits on performance
3. Design provably efficient algorithms



The remainder of the talk

Illustrate

problem structure, fundamental limits, efficient algorithms

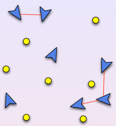
via two scenarios:

1. Distributed Task Allocation
   motivated by a surveillance application
2. Dynamic Vehicle Routing
   motivated by a perimeter defense application

Distributed task allocation

A distributed task allocation problem

- n omnidirectional vehicles
  - limited comm. range and bandwidth
- $m \leq n$ task locations
  - once task is reached by a vehicle, vehicle is forever engaged



Two problem scenarios:

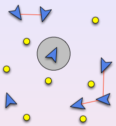
1. Supervisor broadcasts all task locations to each vehicle
2. Vehicles search for task locations with limited range sensor

Problem: distributed algorithm to

- allow group of vehicles to divide tasks among themselves
- minimize time until last task location is reached

A distributed task allocation problem

- n omnidirectional vehicles
  - limited comm. range and bandwidth
- $m \leq n$ task locations
  - once task is reached by a vehicle, vehicle is forever engaged



Two problem scenarios:

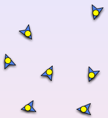
1. Supervisor broadcasts all task locations to each vehicle
2. Vehicles search for task locations with limited range sensor

Problem: distributed algorithm to

- allow group of vehicles to divide tasks among themselves
- minimize time until last task location is reached

A distributed task allocation problem

- n omnidirectional vehicles
  - limited comm. range and bandwidth
- $m \leq n$ task locations
  - once task is reached by a vehicle, vehicle is forever engaged



Two problem scenarios:

1. Supervisor broadcasts all task locations to each vehicle
2. Vehicles search for task locations with limited range sensor

Problem: distributed algorithm to

- allow group of vehicles to divide tasks among themselves
- minimize time until last task location is reached

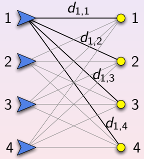
Centralized solution

In the centralized setting, problem is matching in a bipartite graph

Specifically, bottleneck matching: find a matching M which minimizes

$$\max_{M} d_{i,j}$$

Solvable in polynomial time



Distributed challenges

Multi-vehicle task allocation work:

- Auction based (Moore and Passino, 2007)
- Game theoretic (Arslan et al., 2007)
- Auction and consensus (Brunet, Choi and How, 2008)

Today, combination of key challenges:

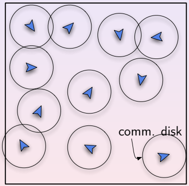
1. range constraint and lack of connectivity
2. tight bandwidth constraint

and novel goals:

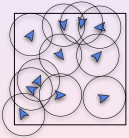
3. determine fundamental limits on scalability
4. develop provably efficient algorithms

Underlying structure: environment size regimes

If # of vehicles increases ($n \to +\infty$)
Then area $A(n)$ must increase to "make room"



Sparse: $A(n)/n \to +\infty$



Dense: $A(n)/n \to 0^+$

Critical: $A(n)/n \to$ constant

Fundamental limits on completion time

Worst-case completion time

- # of tasks = # of vehicles ($m = n$)
- Broadcast or search scenario

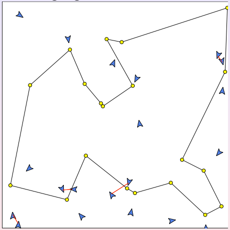
	Sparse ($A(n) \gg n$)	Critical ($A(n) \approx n$)	Dense ($A(n) \ll n$)
Fundamental limit	$\Omega(\sqrt{nA(n)})$	$\Omega(n)$	$\Omega(A(n))$

Asymptotic notation: $T \in \Omega(n)$ implies there is $C > 0$ such that

T lower bounded by Cn

Two allocation algorithms

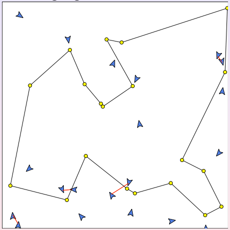
The Ring algorithm



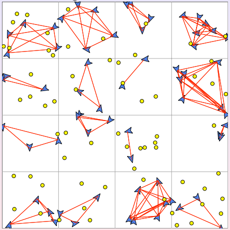
- Compute common ring
- Broadcast scenario

Two allocation algorithms

The Ring algorithm



The Grid algorithm



- Compute common ring
- Broadcast scenario

- Elect leader in each cell
- Broadcast or search

Algorithms match fundamental limit

Worst-case time, (# of tasks m) = (# of vehicles n)

	Sparse	Critical	Dense
Fundamental limit	$\Omega(\sqrt{nA(n)})$	$\Omega(n)$	$\Omega(A(n))$
Ring Alg	$O(\sqrt{nA(n)})$	$O(n)$	$O(\sqrt{nA(n)})$
Grid Alg	$O(A(n))$	$O(n)$	$O(A(n))$

Efficient algorithms

- Ring Alg in sparse and critical environments
- Grid Alg in dense and critical environments

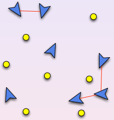
Additional stochastic results have been obtained

Summary of distributed task allocation

Distributed task allocation with communication constraints

The results:

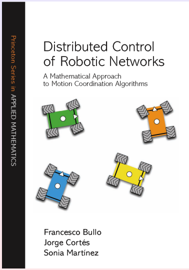
- problem structure: sparse/critical/dense
- fundamental limits on completion time
- efficient algorithms in all three regimes



The technical approach utilizes:

- Distributed algorithms and networking
- Combinatorial optimization
- Random geometric graphs

Text: Distributed Control of Robotic Networks



1. intro to distributed algorithms (graph theory, synchronous networks, and averaging algos)
2. geometric models and geometric optimization problems
3. model for robotic, relative sensing networks, and complexity
4. algorithms for rendezvous, deployment, boundary estimation

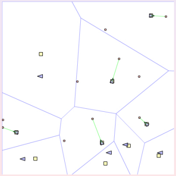
Status: Freely downloadable at http://coordinationbook.info with tutorial slides & software libraries. Shortly on sale by Princeton Univ Press

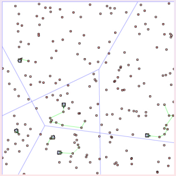
Dynamic vehicle routing

Prior work on dynamic vehicle routing

Dynamic traveling repairperson problem

- Tasks arrive sequentially in time
- Each task location is randomly distributed in service region
- Each task requires on-site service





Key references

Key references

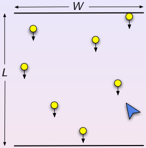
- Shortest path (Beardwood, Halton and Hammersly, 1959)
- Formulation on a graph (Psaraftis, 1988)
- Euclidean plane (Bertsimas and Van Ryzin, 1990–1993)

Recent developments in dynamic vehicle routing:

- Nonholonomic UAVs (Savla, Frazzoli, FB: TAC, (53)6 '08)
- Adaptation and decentralization (Pavone, Frazzoli, FB: TAC, sub '09)
- Distinct-priority targets (SLS, Pavone, FB, Frazzoli: SICON, sub '09)
- Heterogeneous vehicles and teaming (SLS, FB: SCL, sub '08)
- **Moving targets (SBD, SLS, FB: CDC & TAC, sub '09)**

A perimeter defense / boundary guarding problem

- Single vehicle with unit speed
- Task locations (targets):
  - arrive sequentially on a segment
  - move vertically with speed v
- Task completed if target captured before reaching deadline



Goal

Design policies that maximize expected fraction of targets captured

Assume that task arrivals are:

- Poisson in time with rate $\lambda \implies \mathbb{E}[N(\Delta t)] = \lambda \Delta t$
- uniformly distributed on line segment

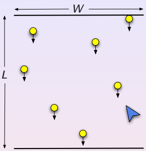
Underlying problem structure

For fixed W, problem parameters are

- speed ratio v:

$$v = \frac{\text{target speed}}{\text{vehicle speed}}$$

- arrival rate λ
- deadline distance L



	$L = +\infty$ Stabilize queue	L is finite Maximize capture fraction
$v < 1$	translational path policy	translational path policy
$v \geq 1$	Not possible for any $\lambda > 0$	longest path policy

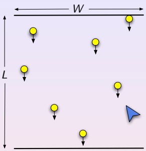
Underlying problem structure

For fixed W, problem parameters are

- speed ratio v:

$$v = \frac{\text{target speed}}{\text{vehicle speed}}$$

- arrival rate λ
- deadline distance L



	$L = +\infty$ Stabilize queue	L is finite Maximize capture fraction
$v < 1$	translational path policy	translational path policy
$v \geq 1$	Not possible for any $\lambda > 0$	longest path policy

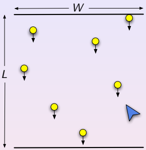
Underlying problem structure

For fixed W, problem parameters are

- speed ratio v:

$$v = \frac{\text{target speed}}{\text{vehicle speed}}$$

- arrival rate λ
- deadline distance L

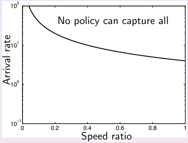


	$L = +\infty$ Stabilize queue	L is finite Maximize capture fraction
$v < 1$	translational path policy	translational path policy
$v \geq 1$	Not possible for any $\lambda > 0$	longest path policy

Fundamental limits for $L = +\infty$ and $v < 1$

For every policy:

$$\lambda \leq \frac{4}{vW}, \quad \text{for stability}$$



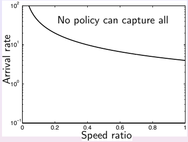
Fundamental limits for $L = +\infty$ and $v < 1$

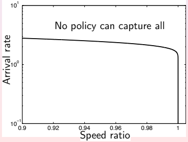
For every policy:

$$\lambda \leq \frac{4}{vW}, \quad \text{for stability}$$

As $v \to 1^-$, for stability

$$\lambda \leq \frac{3\sqrt{2}}{W\sqrt{-\log(1-v)}}$$





Example of proof techniques

For stability, $\lambda \leq \frac{4}{vW}$

1. Distribution of unserviced targets in region of area A:
   - Number is Poisson distributed with parameter $\lambda A/(vW)$
   - Conditioned on number, targets are uniform
2. Targets reachable in time T from (X, Y) are
$$\{(x, y) \mid (X - x)^2 + ((Y - vT) - y)^2 \leq T^2\}$$
3. Probability that closest target is not reachable in T seconds
$$\geq \exp\big(-\lambda\pi T^2/(vW)\big)$$
4. Expected time to travel between targets
$$\mathbb{E}\,[\text{travel time}] \geq \frac{1}{2}\sqrt{\frac{vW}{\lambda}}$$
5. To capture all, $\lambda\mathbb{E}\,[\text{travel time}] \leq 1$

Translational path for $v < 1$

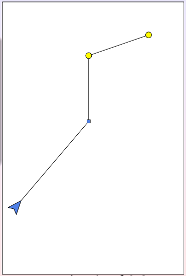
Shortest translational path policy

Input: Optimal location $\mathbf{p}^*$

1. If no targets, then move to $\mathbf{p}^*$
2. Else, capture all targets via shortest translational path
3. Repeat

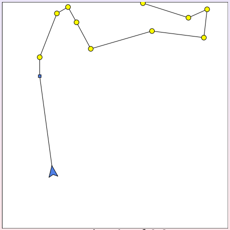
Can compute $\mathbf{p}^*$ to minimize:

- worst-case capture time
- expected capture time



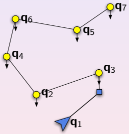
speed ratio of 0.6

Translational path for $v < 1$



speed ratio of 0.2

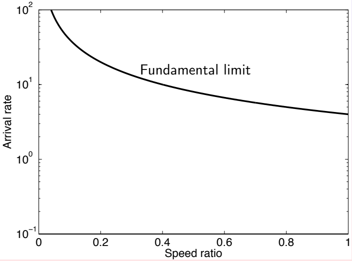
Shortest path computation (Hammar and Nilsson, 2002):



Order: scaled shortest static path
Motion: intercept on straight line

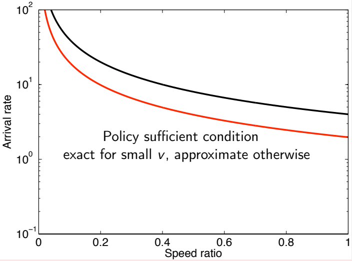
Stability of translational path for $v < 1$

Stability for $L = +\infty$



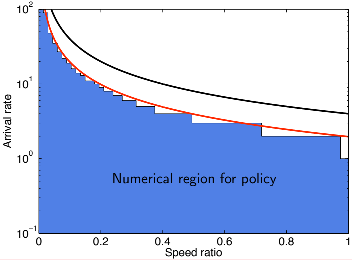
Stability of translational path for $v < 1$

Stability for $L = +\infty$



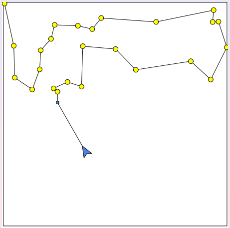
Stability of translational path for $v < 1$

Stability for $L = +\infty$



Maximize capture fraction for $v < 1$

Modify translational path policy



Fundamental limit

$$\text{cap fraction} \leq \min\left\{1, \frac{2}{\sqrt{v\lambda W}}\right\}$$

To analyze policy, assume

- speed ratio v is small
- arrival rate λ is large

Then, capture fraction

$$\geq \min\left\{1, \frac{1.4}{\sqrt{v\lambda W}}\right\}$$

Factor 1.42 of optimal

Numerical results suggest good performance away from limit

Where are we?

	$L = +\infty$ Stabilize queue	L is finite Maximize capture fraction
$v < 1$	translational path policy	modified trans. path policy
$v \geq 1$	Not possible for any $\lambda > 0$	

Where are we?

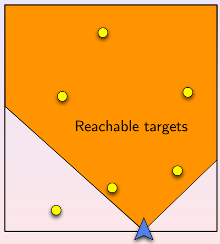
	$L = +\infty$ Stabilize queue	L is finite Maximize capture fraction
$v < 1$	translational path policy	modified trans. path policy
$v \geq 1$	Not possible for any $\lambda > 0$	

Where are we?

	$L = +\infty$ Stabilize queue	L is finite Maximize capture fraction
$v < 1$	translational path policy	modified trans. path policy
$v \geq 1$	Not possible for any $\lambda > 0$	?

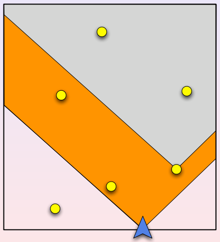
Maximize fraction of targets for $v \geq 1$

For $v \geq 1$, it is optimal to remain on deadline



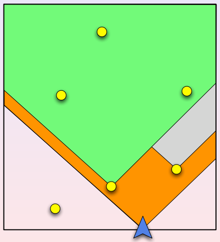
Maximize fraction of targets for $v \geq 1$

For $v \geq 1$, it is optimal to remain on deadline



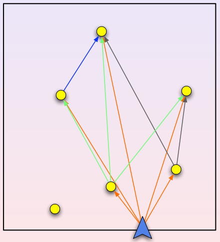
Maximize fraction of targets for $v \geq 1$

For $v \geq 1$, it is optimal to remain on deadline



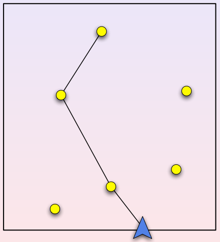
Maximize fraction of targets for $v \geq 1$

For $v \geq 1$, it is optimal to remain on deadline



Maximize fraction of targets for $v \geq 1$

For $v \geq 1$, it is optimal to remain on deadline



- Reachability graph is directed and acyclic

Fundamental limit for $v \geq 1$

Noncausal information $=$ *a priori* knowledge of
arrival time and location of every future target

Optimal performance with noncausal information

1. Compute infinite reachability graph of all future targets
2. Compute longest path in graph
3. Capture each target on path

Consequences for algorithm performance (capture fraction)

- noncausal performance can be computed
- noncausal performance is upper bound on causal performance

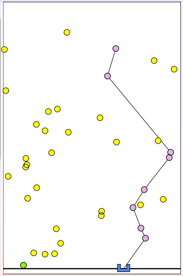
Capture fraction with $v \geq 1$: Longest path policy

Longest path (LP) policy

1. Compute the reachability graph of all unserviced targets
2. Compute longest path in graph
3. Capture first target on path by intercepting on deadline
4. Repeat

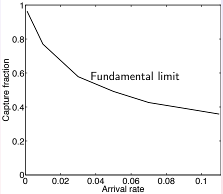
Capture fraction for $L > vW$:

Factor $\left(1 - \frac{vW}{L}\right)$ of optimal

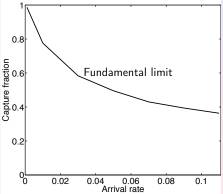


Numerical capture fraction for $v \geq 1$

Environment with $W = 2$ and $L = 5$.



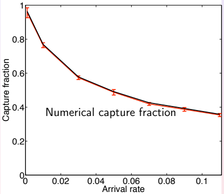
$v = 2$ and thus $L > vW$



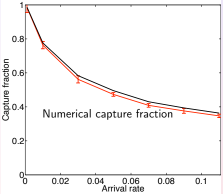
$v = 5$ and thus $L < vW$

Numerical capture fraction for $v \geq 1$

Environment with $W = 2$ and $L = 5$.



$v = 2$ and thus $L > vW$



$v = 5$ and thus $L < vW$

Summary of boundary guarding

The results:

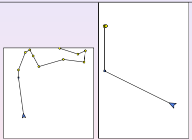
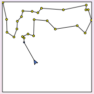
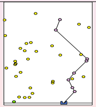
- Identified four regimes
- Derived fundamental limits on capture fraction
- Developed provably efficient algorithms

	$L = +\infty$ Stabilize queue	L is finite Maximize capture fraction
$v < 1$	translational path policy	translational path policy
$v \geq 1$	Not possible for any $\lambda > 0$	longest path policy

The technical approach utilizes:

- Stochastic processes and queueing
- Combinatorial optimization

Summary of boundary guarding: policies

	$L = +\infty$ Stabilize queue	L is finite Maximize capture fraction
$v < 1$		
$v \geq 1$	Not possible for any $\lambda > 0$	

Summary

Future autonomous missions

- Fleets (swarms) of networked vehicles
- Complex sets of tasks that evolve during execution
- Increased autonomy, humans as supervisors

Enabling technology: real-time task allocation and vehicle routing

Technical approach: Fundamental theory and algorithms

1. underlying problem structure
2. fundamental limits on performance
3. simple, provably efficient algorithms