Forward Search Value Iteration For POMDPs
IJCAI’07
Guy Shani & Rohen I. Brafman & Solomon E. Shimony
Computer Science Department . Ben-Gurion University, Israel

Gilles S. Dibangoye
Computer Science Department
Laval University

September 21, 2007
Primary motivations of the paper

POMDP problem (recall)
- The Model
- POMDP challenges

State of the art POMDP Algorithms
- Point based solvers
- HSVI Algorithm

FSVI Algorithm
- The Algorithm
- Experimental Framework
Primary motivations of the paper

1. **Scale up better on larger POMDP domains:**
   - Rocksparse(8, 8) (where $|S| = 16,385$, $|A| = 13$ and $|O| = 2$);

2. **Improve the performances of the state-of-the-art HSVI:**
   - Convergence rate;

3. **Suggest a new framework and metrics for fair comparisons among algorithms:**
   - The estimate of the **Average Discounted Reward (ADR);**
   - The execution time metrics (e.g., the number of backups);
   - The required memory (e.g., the size of the value function);
Outline

1. Primary motivations of the paper

2. POMDP problem (recall)
   - The Model
   - POMDP challenges

3. State of the art POMDP Algorithms
   - Point based solvers
   - HSVI Algorithm

4. FSVI Algorithm
   - The Algorithm
   - Experimental Framework
MDP and POMDP models

**MDP model** $\langle S, A, T, R, \gamma \rangle$

1. $S$: the finite set of states $s$;
2. $A$: the finite set of actions $a$;
3. $T$: the transition function where $T(s'|s, a)$ denotes the probability of transiting from $s$ to $s'$ after executing action $a$;
4. $R$: the reward function that defines the reward $R(s, a) \in \mathbb{R}$ of executing $a$ in state $s$;
5. $\gamma$: discount factor.

**POMDP model** $\langle S, A, T, R, \gamma, \Omega, O, b_0 \rangle$ extends the MDP model:

1. $\Omega$: the finite set of observations $o$;
2. $O$: the observation function where $O(o|s', a)$ is the probability of observing $o$ after executing $a$ and reaching state $s'$.
3. $b_0$: the initial probability distribution among states $s \in S$. 
Outline

1. Primary motivations of the paper

2. POMDP problem (recall)
   - The Model
   - POMDP challenges

3. State of the art POMDP Algorithms
   - Point based solvers
   - HSVI Algorithm

4. FSVI Algorithm
   - The Algorithm
   - Experimental Framework
POMDP challenges

Objective function

POMDP exact solvers aim at finding $V^*$ such that: $\forall b \in B$

- **Bellman Equation:**

$$V^*(b) = \max_{a \in A} \left( R(b, a) + \gamma \sum_{o \in \Omega} P(o|b, a)V^*(\tau(b, a, o)) \right) \quad (1)$$

- where $\tau$ is the beliefs’ transition function: $\forall b \in B, a \in A$ and $o \in \Omega$:

$$\tau(b, a, o) = b'$$

$$b'(s') = \eta O(o|s', a) \sum_{s \in S} T(s'|s, a)b(s)$$

- and $B$ denotes the continuous set of beliefs $b$. 
Outline

1. Primary motivations of the paper

2. POMDP problem (recall)
   - The Model
   - POMDP challenges

3. State of the art POMDP Algorithms
   - Point based solvers
   - HSVI Algorithm

4. FSVI Algorithm
   - The Algorithm
   - Experimental Framework
Point based solvers: commonly shared ideas

1. Constrain the belief space $B$, by planning only on a finite set of reachable beliefs.
2. Use point-based backup instead of exhaustive backup performed by exact solution methods.

Advantages and Limitations

- ✔ The backup operator is polynomial instead of exponential;
- ✔ Point based algorithms scale up to medium-size POMDPs;
- ✗ The set of reachable beliefs may be very large;
- ✗ Point based solvers do not focus on the most relevant parts of the reachable beliefs, they proceed mainly in a synchronous manner.
Outline

1. Primary motivations of the paper

2. POMDP problem (recall)
   - The Model
   - POMDP challenges

3. State of the art POMDP Algorithms
   - Point based solvers
   - HSVI Algorithm

4. FSVI Algorithm
   - The Algorithm
   - Experimental Framework
HSVI - An overview

- HSVI uses the point-based backup operator;
- It maintains upper $\bar{V}$ and lower $\underline{V}$ bounds on the value function;
- It selects the next belief $b' = \tau(b, a^\star, o^\star)$ to be backed up according to:
  1. The current belief $b$;
  2. The best action $a^\star$ according to $\bar{V}$:

$$a^\star = \arg \max_a Q_{\bar{V}}(b, a) \quad (2)$$

  3. The observation $o^\star$ that minimize the gap between $\bar{V}$ and $\underline{V}$:

$$o^\star = \arg \max_o \bar{V}(\tau(b, a^\star, o)) - \underline{V}(\tau(b, a^\star, o)) \quad (3)$$

Limitations

- Operations Equation (2) and (3) are quite expensive.
Outline

1. Primary motivations of the paper

2. POMDP problem (recall)
   - The Model
   - POMDP challenges

3. State of the art POMDP Algorithms
   - Point based solvers
   - HSVI Algorithm

4. FSVI Algorithm
   - The Algorithm
   - Experimental Framework
Objectives
Find sequences of useful backups that tradeoff between both:
1. Gather rewards and information;
2. Do not require a considerable computational effort.

Trial based solver
1. Generate belief trajectories according to a policy (heuristic or not);
   - Direct the trajectories towards the rewards;
   - Sample the next belief according to the heuristic or current policy.
2. Back up beliefs in the reverse order of the trajectories.
   - This principle is commonly used in RTDP-style algorithms.
FSVI - the algorithm

FSVI

- Initialize $V$
  - While ($V$ has not converged)
    - sample $s_0$ from the $b_0$ distribution
    - MDPExplore($b_0, s_0$)
  - EndWhile

- MDPExplore($b, s$)
  - If ($s$ is not a goal state)
    - $a^* \leftarrow \arg \max_{a'} Q_{MDP}(s, a')$
    - sample $s'$ from $P(s, a^*, \star)$
    - sample $o$ from $O(a^*, s', \star)$
    - MDPExplore($\tau(b, a^*, o), s'$)
    - add($V, \text{backup}(b, V)$)
  - EndIf
Illustration example

Figure: Reward traces of the $Q_{MDP}$. (Y. Viring, G. Shani et al. AAAI’07)
Outline

1. Primary motivations of the paper

2. POMDP problem (recall)
   - The Model
   - POMDP challenges

3. State of the art POMDP Algorithms
   - Point based solvers
   - HSVI Algorithm

4. FSVI Algorithm
   - The Algorithm
   - Experimental Framework
Evaluation metrics

1. **Value function evaluation.** Average discounted reward:

\[
\frac{\sum_{i=0}^{\text{#trials}} \sum_{j=0}^{\text{#steps}} \gamma^j r_j}{\text{#trials}}
\]

where #trials stands for the number or trials, and #steps denotes the horizon.

2. **CPU time.** The amount of basics operations:
   - the number of \(\tau\) operations
   - the number of dot products \(\alpha \cdot b\)
   - the number of backups ...

3. **Memory.** The size of the computed value function.
FSVI has been evaluated on most POMDP benchmarks including rocksample domains.

**The rocksample problem (HSVI - T. Smith et al. 2005)**

*Figure:* Instance of RockSample domain: RockSample[7, 8]
Figure: Normalized comparison of CPU time for Rocksamp problems
Figure: Convergence rate on noisy Rocksamle 8,8 problem.
### Results (3)

| Method       | ADR | $|V|$   | Time (secs) | # Backups | $g_{q,a}^{+}$ x 10^6 | $|B|$ x 10^4 | $|\tau|$ x 10^3 | $\alpha \cdot b$ | $|V|$ x 10^3 | $V(b)$ | $V(b)$ | #IV |
|--------------|-----|--------|-------------|-----------|------------------------|-------------|--------------|-------------|-------------|---------|---------|------|
| Hallway      | 0.516 | 182 | 314 | 6.34 | 5.85 | 3.4 | 34.52 | 6.67 | 0.42 | 106.1 | 1268 | 0.57μs | 65ms |
| FSVI         | 0.517 | 233 | 50  | 6.55 | 7.71 | 0.05 | 0.51 | 7.78 | 0.08 | 100.9 | 1268 | 0.57μs | 65ms |
|              | ±0.0024 | ±71 | ±18 | ±1.00 | ±2.46 | ±0.01 | ±0.08 | ±2.49 | 0.08 | 106.1 | 1268 | 0.57μs | 65ms |
| Hallway2     | 0.516 | 172 | 99  | 217 | 1.56 | 2.11 | 11.07 | 1.81 | 0.23 | 37.2 | 434   | 0.57μs | 65ms |
| FSVI         | 0.515 | 296 | 49  | 355 | 4.28 | 0.03 | 0.3  | 4.33 | 0.3   | 100.9 | 1268 | 0.57μs | 65ms |
|              | ±0.0027 | ±22 | ±8  | ±3.3 | ±0.73 | ±0.03 | ±0.74 | 0.08 | 106.1 | 1268 | 0.57μs | 65ms |
| Tag Avoid    | 0.518 | 100 | 52  | 304 | 0.5  | 0.29 | 1.74 | 0.53 | 1.1   | 29.3 | 1635 | 0.57μs | 65ms |
| FSVI         | 0.517 | 174 | 45  | 182 | 0.37 | 0.01 | 0.14 | 0.39 | 0.3   | 100.9 | 1268 | 0.57μs | 65ms |
|              | ±0.015 | ±25 | ±8  | ±27 | ±0.1 | ±0.02 | ±0.11 | 0.08 | 106.1 | 1268 | 0.57μs | 65ms |
| Rock Sample 4,4 | 0.518 | 103 | 4   | 207 | 1.08 | 0.1 | 1.17 | 1.09 | 0.34 | 6.06 | 414   | 0.57μs | 65ms |
| FSVI         | 0.519 | 84  | 1   | 204 | 0.24 | 0  | 0.04 | 0.25 | 0.3   | 100.9 | 1268 | 0.57μs | 65ms |
|              | ±0.024 | ±76 | ±1  | ±122 | ±0.41 | ±0.01 | ±0.42 | 0.08 | 106.1 | 1268 | 0.57μs | 65ms |
| Rock Sample 5,5 | 0.518 | 114 | 83  | 2309 | 10.39 | 0.26 | 2.34 | 10.5 | 0.8  | 101.1 | 1883 | 0.57μs | 65ms |
| FSVI         | 0.518 | 279 | 11.1 | 6262 | 1.47 | 0.02 | 0.1 | 1.49 | 0.3   | 100.9 | 1268 | 0.57μs | 65ms |
|              | ±0.063 | ±75 | ±5  | ±247 | ±0.88 | ±0.02 | ±0.89 | 0.08 | 106.1 | 1268 | 0.57μs | 65ms |
| Noisy Rock Sample 5,5 | 0.518 | 132 | 83  | 3108 | 0.86μs | 0.035μs | 0.64μs | 24.1μs | 0.64μs | 24.1μs | 0.64μs | 24.1μs |
| FSVI         | 0.517 | 2639 | 1.586 | 3528 | 1.329 | 2.01 | 23.05 | 12.4 | 0.88 | 264.9 | 9294 | 0.57μs | 65ms |
|              | ±0.069 | ±170 | ±52 | ±224 | ±4.79 | ±0.12 | ±4.8 | 0.08 | 106.1 | 1268 | 0.57μs | 65ms |
| Rock Sample 5,7 | 0.518 | 145 | 205 | 350 | 0.65 | 1 | 4.3 | 0.99 | 3.44 | 14.09 | 702 | 0.57μs | 65ms |
| FSVI         | 0.518 | 306.9 | 34.3 | 500 | 39648 | 3722 | 0.4  | 2.1 | 0.4   | 100.9 | 1268 | 0.57μs | 65ms |
|              | ±0.63 | ±91.5 | ±13.6 | ±400.1 | ±0.02 | ±0.014 | ±2.0 | 0.46 | 100.9 | 1268 | 0.57μs | 65ms |
| Rock Sample 7,8 | 0.518 | 567 | 20μs | 205 | 14.51 | 1.5 | 4.1 | 14.66 | 3.42 | 9.53 | 628 | 0.57μs | 65ms |
| FSVI         | 0.518 | 343.1 | 239 | 512.1 | 2.389 | 0.049 | 0.024 | 2.457 | 0.46 | 100.9 | 1268 | 0.57μs | 65ms |
|              | ±0.265 | ±146.6 | ±78.7 | ±284.8 | ±2.392 | ±0.059 | ±2.357 | 0.46 | 100.9 | 1268 | 0.57μs | 65ms |
| Rock Sample 8,8 | 0.518 | 570 | 25μs | 25μs | 0.35μs | 2.6μs | 18μs | 2.7μs | 0.35μs | 2.6μs | 18μs | 2.7μs | 0.35μs | 2.6μs |
| FSVI         | 0.518 | 762 | 13917 | 1317 | 10.83 | 4.01 | 58.09 | 11.43 | 17.3 | 58.38 | 2638 | 0.57μs | 65ms |
|              | ±0.337 | ±76.9 | ±205 | ±125.1 | ±0.02 | ±0.013 | ±0.06 | ±0.66 | 0.46 | 100.9 | 1268 | 0.57μs | 65ms |
| Network Ring 8 | 0.518 | 164 | 1μs | 2.6μs | 4.2μs | 11μs | 31.8μs | 0.57μs | 65ms |
| FSVI         | 0.518 | 83 | 19 | 153 | 0.004 | 0.25 | 8.44 | 0.31 | 0.39 | 8.46 | 307 | 0.57μs | 65ms |
|              | ±0.18 | ±16 | ±6.1 | ±146 | ±0.002 | ±0.012 | ±0.14 | ±0.19 | 0.39 | 8.46 | 307 | 0.57μs | 65ms |
| Network Ring 10 | 0.518 | 553 | 10.6μs | 13.3μs | 23.3μs | 99.4μs | 369μs | 0.57μs | 65ms |
| FSVI         | 0.518 | 69 | 141 | 103 | 0.0036 | 0.29 | 6.9 | 0.144 | 1.11 | 6.9 | 206 | 0.57μs | 65ms |
|              | ±0.03 | ±6.14 | ±14.3 | ±84.78 | ±0.0008 | ±0.008 | ±0.81 | ±0.086 | 0.39 | 8.46 | 307 | 0.57μs | 65ms |

**Figure:** Performance measurements on different domains.
Why does FSVI work?

Main insights

- By using $Q_{\text{MDP}}$ heuristic, FSVI directs the trajectories towards the belief sub-space concentrated on states handling the highest values of the $Q_{\text{MDP}}$.
- Thus by backing up such beliefs, it considerably improves the quality of the value function.
- The belief expansion step is not that expensive, thus generating such good trajectories does not require a great amount of computational effort.
### Limitations of FSVI

- **X** Inability of $Q_{\text{MDP}}$ heuristic to perform information gathering tasks.
  - **The tiger** problem: `listen` action is just forgotten in the $Q_{\text{MDP}}$.
  - **The heaven-hell** problem: requires long-term information gathering.

- **X** FSVI may loop when the underlying MDP is a **weakly communicating** MDP with absorbing state subspaces.

### Open questions!!

1. Is FSVI sequences the ones that achieve the best tradeoff?
2. Is there any way to generate such trajectories backwards?
3. How to tackle the information gathering problem in FSVI?