*Bioinformatics*, 34(5), 2018, 770–778
doi: 10.1093/bioinformatics/btx638
Advance Access Publication Date: 11 October 2017
Original Paper

OXFORD

Structural bioinformatics

# Machine learning accelerates MD-based binding pose prediction between ligands and proteins

**Kei Terayama[1,*], Hiroaki Iwata[2], Mitsugu Araki[3], Yasushi Okuno[3,4] and Koji Tsuda[1,5,6,*]**

[1]Department of Computational Biology and Medical Science, Graduate School of Frontier Sciences, The University of Tokyo, Chiba 277-8561, Japan, [2]Foundation for Biomedical Research and Innovation, Hyogo 650-0047, Japan, [3]RIKEN Advanced Institute for Computational Science, Hyogo 650-0047, Japan, [4]Department of Biomedical Data Intelligence, Graduate School of Medicine, Kyoto University, Kyoto 606-8507, Japan, [5]Center for Materials Research by Information Integration, NIMS, Ibaraki 305-0047, Japan and [6]RIKEN Center for Advanced Intelligence Project, Tokyo 103-0027, Japan

*To whom correspondence should be addressed.
Associate Editor: Burkhard Rost

Received on May 1, 2017; revised on September 11, 2017; editorial decision on October 5, 2017; accepted on October 10, 2017

## Abstract

**Motivation:** Fast and accurate prediction of protein–ligand binding structures is indispensable for structure-based drug design and accurate estimation of binding free energy of drug candidate molecules in drug discovery. Recently, accurate pose prediction methods based on short Molecular Dynamics (MD) simulations, such as MM-PBSA and MM-GBSA, among generated docking poses have been used. Since molecular structures obtained from MD simulation depend on the initial condition, taking the average over different initial conditions leads to better accuracy. Prediction accuracy of protein–ligand binding poses can be improved with multiple runs at different initial velocity.
**Results:** This paper shows that a machine learning method, called Best Arm Identification, can optimally control the number of MD runs for each binding pose. It allows us to identify a correct binding pose with a minimum number of total runs. Our experiment using three proteins and eight inhibitors showed that the computational cost can be reduced substantially without sacrificing accuracy. This method can be applied for controlling all kinds of molecular simulations to obtain best results under restricted computational resources.
**Availability and implementation:** Code and data are available on GitHub at https://github.com/tsudalab/bpbi.
**Contact:** terayama@cbms.k.u-tokyo.ac.jp or tsuda@k.u-tokyo.ac.jp
**Supplementary information:** Supplementary data are available at *Bioinformatics* online.

## 1 Introduction

Structure-based drug discovery is becoming an essential tool for assisting fast and cost-efficient lead discovery and optimization (Lionta *et al.*, 2014). Fast and accurate prediction of protein–ligand binding structure is an indispensable process for structure-based drug design, especially when binding structure has not been resolved by X-ray or NMR. In addition, accurate prediction of binding structure enables us to estimate binding free energy $\Delta G_{bind}$ with high accuracy using existing methods such as Linear Interaction Energy (Åqvist *et al.*, 1994), Molecular Mechanics/Poisson–Boltzmann Surface Area (MM-PBSA) (Kollman *et al.*, 2000; Srinivasan *et al.*, 1998), Molecular Mechanics/Generalized Born Surface Area (MM-GBSA) (Onufriev *et al.*, 2000) or Massively Parallel Computation of Absolute binding Free Energy with well-Equilibrated states (Fujitani *et al.*, 2009). Docking programs are often used as scoring function for binding pose prediction, but their accuracy is still low (Hou *et al.*, 2011; Lavecchia

and Di Giovanni, 2013). Recently, MM-PBSA-based binding pose prediction methods have also been proposed, in which Molecular Dynamics (MD) and MM-PBSA calculations are performed to estimate $\Delta G_{\mathrm{bind}}$s on generated docking poses and accurately identify the best ones (Hou *et al.*, 2011; Thompson *et al.*, 2008). A number of methods for binding pose prediction and binding free energy estimation based on MD calculation have been proposed (Colizzi *et al.*, 2010; Okimoto *et al.*, 2009; Proctor *et al.*, 2012).

However, binding pose prediction based on the MM-PBSA method is computationally expensive: in order to improve accuracy, MD simulation and MM-PBSA calculation (MD and MM-PBSA run) must be repeated and averaged over multiple initialization conditions, for each pose candidate (Berhanu and Hansmann, 2013; Genheden and Ryde, 2010; Mikulskis *et al.*, 2012; Sadiq *et al.*, 2010). This approach incurs a huge computational cost, as existing studies use the same number of initial conditions for all poses (uniform sampling) and unnecessary calculation is performed for unpromising pose candidates. Figure 1A shows the pose prediction with MD and MM-PBSA through uniform sampling.

The recently introduced Best Arm Identification (BAI) problem consists of optimizing the allocation of limited resources to find the best slot out of many slots. In this problem, we select a slot, called arm, and get a reward according to a probability distribution associated with it. These probability distributions are not known to us. The purpose of the problem is to minimize the total number of selections and reward-getting processes to find the best arm. A number of effective algorithms to achieve this purpose have been proposed (Audibert and Bubeck, 2010; Bubeck *et al.*, 2009; Gabillon *et al.*, 2012).

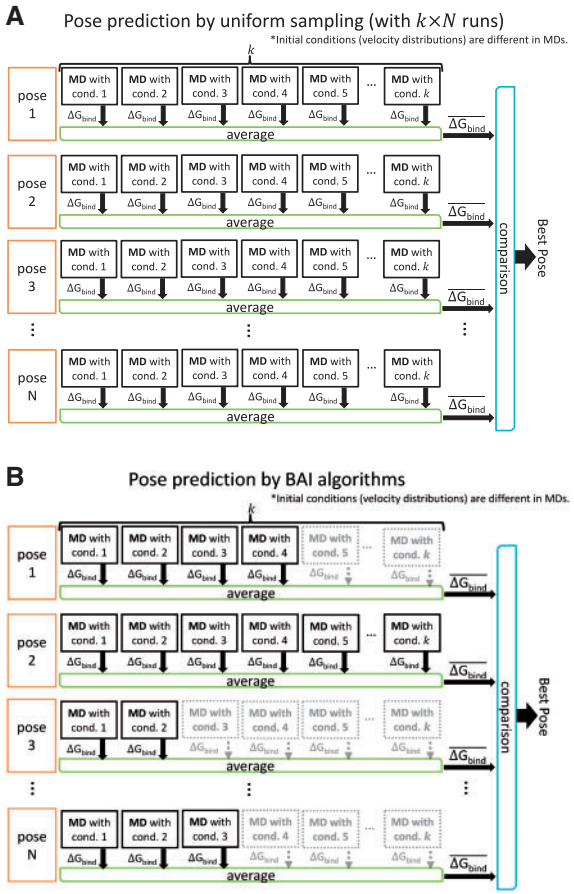


**Fig. 1.** Pose prediction using uniform sampling (**A**) and BAI (**B**) algorithms. The purpose of pose prediction is to select the best (minimum $\overline{\Delta G_{20\mathrm{vel.}}}$) pose among $N$ prepared docking poses. Using uniform sampling, the same number ($k$) of MD and MM-PBSA runs with different initial velocities is performed, resulting in a total of $k \times N$ runs. On the other hand, the total number of runs can be reduced by optimally controlling runs using a BAI algorithm (**B**) (Color version of this figure is available at *Bioinformatics* online.)

The BAI problem has attracted attention in the field of machine learning (Auer *et al.*, 2002; Bubeck *et al.*, 2009) and has been applied to various fields such as the design of clinical trials (Villar *et al.*, 2015), recommendation systems of news and goods (Li *et al.*, 2010) and the game of Go (Coulom, 2006; Silver *et al.*, 2016). In this paper, we propose an effective pose prediction method using a BAI algorithm. Using such an algorithm to optimally control MD and MM-PBSA runs, we can reduce the total number of MD and MM-PBSA runs compared to the total number of runs with uniform sampling, as seen in Figure 1.

To show the effectiveness of the proposed method, we conducted a pose prediction experiment, using the MD and MM-PBSA methods, on a dataset consisting of three proteins, cyclin-dependent kinase 2 (CDK2), heat shock protein 90 alpha (HSP90A) and coagulation factor X (FA10), and eight inhibitors. We prepared 20 binding poses that included one or more 'correct' poses for each complex. We then investigated how much of a computational cost reduction can be obtained on runs of MD and MM-PBSA, using our proposed method, compared with the widely used uniform sampling approach (where the same number of MD and MM-PBSA runs is performed for each pose).

The number of MM-PBSA runs was reduced by a factor of 1.76–6.67 compared to uniform sampling, with a success probability of correct pose identification fixed to 95%. In particular, in cases where pose prediction was difficult due to the small difference between correct and incorrect poses, $\Delta G_{\mathrm{bind}}$, the cost reduction was even greater. This result illustrates the ability of BAI algorithms to avoid unpromising poses at an early stage and concentrate computation resources on promising poses instead.

## 2 Materials and methods

### 2.1 Binding pose prediction and BAI

In order to predict binding poses, we need to estimate and compare the binding free energies, $\Delta G_{\mathrm{bind}}$s, of each generated pose, as shown in Figure 1. An accurate value of $\Delta G_{\mathrm{bind}}$ for a pose can be obtained by performing MD and MM-PBSA runs with multiple initial velocities. Uniform sampling, in which the same number of runs with different initial conditions is conducted, is the current standard method to obtain the best results (Berman *et al.*, 2000; Genheden and Ryde, 2010; Mikulskis *et al.*, 2012; Sadiq *et al.*, 2010), however it comes with enormous calculation costs (Fig. 1A).

We propose an effective pose prediction method that controls runs based on BAI. The BAI problem was formulated by Bubeck *et al.* (2009) in the field of machine learning. Figure 2A shows a flowchart of the BAI algorithm in general settings. The purpose of the BAI algorithm is to optimize allocation of limited resources to find the best slot (arm). At a first step, a forecaster pulls each arm once and observes a sample drawn from the reward distribution of the arm (Initialization). He repeatedly selects an arm according to the 'scores' of the arms (Selection) and get a reward of the arm, until the budget runs out (Pull). The selection way of arms and the definition of 'score' depends on exploration algorithms. Finally, he selects the seemingly best arm from the reward of each arm (Final Selection). Since he does not know the reward distributions, he needs to explore entire arms and exploits the seemingly most

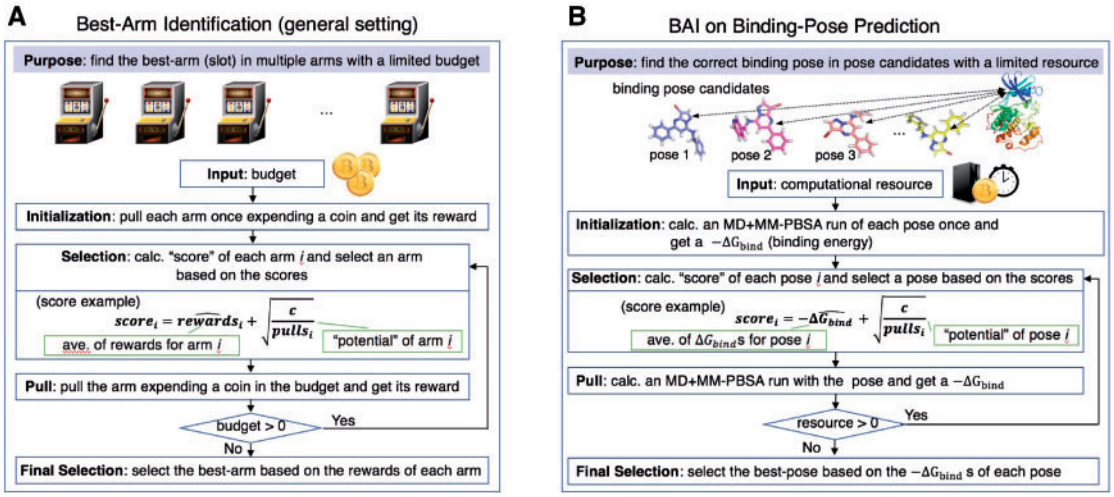


**Fig. 2.** The basic idea for our framework. (**A**) A flowchart of the BAI algorithm in the general setting. The purpose is to find the best arm (slot) by repeating selection and reward acquisition within a limited budget. (**B**) The BAI algorithm applied to the binding pose prediction problem. We can reduce the total number of MD and MM-PBSA runs to find the binding pose by efficient BAI algorithms (Color version of this figure is available at *Bioinformatics* online.)

rewarding arms, avoiding the seemingly bad rewarding arms. A number of effective algorithms for the BAI problem have been proposed in some settings, and theoretical analysis has been also conducted (Audibert and Bubeck, 2010; Bubeck *et al.*, 2009; Gabillon *et al.*, 2012; Kaufmann and Kalyanakrishnan, 2013).

The problem of binding pose prediction can be regarded as a BAI problem as shown in Figure 2B by considering an arm as a binding pose, pulling an arm as calculation of $-\Delta G_{\text{bind}}$ by MD and MM-PBSA, and a reward as an estimate $-\Delta G_{\text{bind}}$. [In this study, we multiplied MM-PBSA estimates by − 1 before applying BAI algorithms. This is because BAI algorithms output the best (maximum) arm, whereas the binding pose that has smallest binding energy should be explored.] BAI algorithms can reduce the total number of MD and MM-PBSA runs because they optimally control the number of runs for each pose to identify the binding pose.

### 2.1.1 BAI algorithms used in our experiment

We adopt four BAI algorithms for our experiments: (i) Upper Confidence Bound algorithm with quantile factor parameter $p$ (UCB($p$)) (Bubeck *et al.*, 2009), (ii) Upper Confidence Bound Exploration (UCB-E) (Audibert and Bubeck, 2010), (iii) Successive Rejects (SR) (Audibert and Bubeck, 2010) and (iv) Unified Gap-based Exploration (UGapE) (Gabillon *et al.*, 2012). Their effectiveness has been shown both theoretically and experimentally, and implementation is relatively easy.

To explain the details of these algorithms, we introduce standard BAI notation (see overview in Fig. 3): Let $A = \{1, 2, \ldots, K\}$ be the set of arms and $n$ the number of rounds (or budget). For $t = 1, 2, \ldots, n$, at round $t$, the forecaster chooses an arm $I_t$ in $A$ and observes a reward. A reward of an arm $i$ is sampled from the reward distribution $\nu_i$, which is the unknown parameter for the forecaster. For each arm $i$ and round $t$, we denote by $T_i(t)$ the number of times arm $i$ was pulled from rounds 1 to $t$, and by $X_{i,1}, X_{i,2}, \ldots, X_i T_i(t)$ the sequence of associated rewards. We define the empirical mean for arm $i$ after $s$ pulls $\widehat{X}_{i,s} = \frac{1}{s}\Sigma_{t=1}^{s} X_{i,t}$. After $n$ rounds, the forecaster selects an arm, denoted by $J_n$.

The UCB($p$) and UCB-E algorithms were proposed by Bubeck *et al.* (2009) and Audibert and Bubeck (2010), respectively, based off the UCB algorithm (Auer *et al.*, 2002), which is practical and widely used for the multi-armed bandit (MAB) problem. The MAB problem is a problem to optimize allocation of limited resources based on existing knowledge in order to maximize the cumulative sum of rewards obtained by multiple slots (arms) (Agrawal and Goyal, 2012; Auer *et al.*, 2002; Robbins, 1952). The BAI problem is a variant of the MAB problem. Algorithms 1 and 2 show the UCB($p$) and the UCB-E algorithms. The basic idea of these algorithms is to select the most promising arm

Parameters available to the forecaster: the number of rounds $n$ and the number of arms $K$.
Parameters unknown to the forecaster: the reward distributions $\nu_1, \nu_2, \ldots, \nu_K$ of the arms.
For each round $t = 1, 2, \ldots, n$:
(1) the forecaster chooses an arm $I_t \in \{1, 2, \ldots, K\}$.
(2) the environment draws the reward $X_{I_t, T_{I_t}(t)}$ from the arm $I_t$.
At the end of the $n$ rounds, the forecaster outputs a recommendation $J_n \in \{1, 2, \ldots, K\}$

**Fig. 3.** Problem formulation of BAI

**Algorithm 1** UCB($p$) algorithm.

**Require:** $p > 0$ (exploration parameter), $n$ (total number of rounds)
**for** $i \leftarrow 1, K$ **do** ▷ Initialization
  Pull arm $i$ and get reward $X_{i,1}$
**end for**
**for** $t \leftarrow K+1, n$ **do** ▷ Exploration loop
  **for** $i \leftarrow 1, K$ **do** ▷ Calculate UCB($p$) score $S_i$ of arm $i$
    $S_i \leftarrow \widehat{X}_{i,T_i(t-1)} + \sqrt{\frac{p \log(t-1)}{T_i(t-1)}}$
    ▷ $T_i(t-1)$: the number of times arm $i$ was pulled until $t-1$
  **end for**
  Draw arm $I_t \in \arg\max_{i \in A} S_i$ and get reward $X_{I_t, T_{I_t}(t-1)+1}$
**end for**
**return** $J_n \in \arg\max_{i \in A} \widehat{X}_{i,T_i(n)}$

**Algorithm 2** UCB-E algorithm.

**Require:** $c > 0$ (exploration parameter), $n$ (total number of rounds)
1: **for** $i \leftarrow 1, K$ **do** ▷ Initialization
2: Pull arm $i$ and get reward $X_{i,1}$
3: **end for**
4: **for** $t \leftarrow K+1, n$ **do** ▷ Exploration loop
5: **for** $i \leftarrow 1, K$ **do** ▷ Calculate UCB-E score $S_i$ of arm $i$
6: $S_i \leftarrow \widehat{X}_{i,T_i(t-1)} + \sqrt{\frac{c}{T_i(t-1)}}$
7: **end for**
8: Draw arm $I_t \in \arg\max_{i \in A} S_i$ and get reward $X_{I_t, T_{I_t}(t-1)+1}$
9: **end for**
10: **return** $J_n \in \arg\max_{i \in A} \widehat{X}_{i,T_i(n)}$

based on each score function. These algorithms approximately calculate upper confidence bounds (UCBs) of arms according to some criteria and select an arm based on the UCBs. When using these methods, it is necessary to decide the total number of rounds (budget) $n$ and the exploration parameters $p$ and $c$ beforehand. In general, when the value of an exploration parameter is large, it is extensively searched and when the value is small, a small number of candidates are intensively searched. Users need to determine a proper value of the parameter to find the best arm effectively without missing it.

Audibert et al. also proposed the SR algorithm (Algorithm 3), which is an exploration parameter free algorithm. The basic idea of SR is to reject unpromising arms and pull the remaining arms. SR has the same number of phases as the number of arms and discards one arm, which is seemingly bad per phase. The number of poses calculated in each phase $k$ is determined by $n_k$ in Algorithm 3. The sum of $n_k$ for $k = 1, 2, \ldots, K-1$ is less than or equal to $n$. We also implemented UGapE algorithm (Algorithm 4), which outperformed existing algorithms including UCB-E in some settings (Gabillon *et al.*, 2012). UGapE was proposed as an algorithm to find the best $m$ arms. We fix $m = 1$ in this paper. In the selection process of UGapE, we first find the 'promising' arm $l_t$ based on a confident interval $\beta_k(t-1)$ and then select an arm among the arm $l_t$ and the 'best' one $u_t$ in the arms except for $l_t$. This algorithm also uses the exploration parameter $a$.

It is difficult to predict which algorithm is suitable for a given problem in advance, because the effectiveness of exploration depends on the distribution of rewards on the problem and exploration parameters. We experimentally verify which algorithm is appropriate for the binding pose prediction problem.

#### 2.1.2 Automatic estimation of exploration parameters

One practical issue with the algorithms covered in this paper is the selection of exploration parameters that affect the effectiveness of exploration and are needed to be determined by users. We employ automatic adjustment methods for the free parameters $c$ and $a$ in UCB-E and UGapE, as stated by Audibert and Bubeck (2010) and Gabillon *et al.* (2011).

For these two algorithms with reward distributions over [0, 1], the bounds of probability of error $e_n$, i.e. the probability that the finally selected arm $J_n$ is not the best one with round $n$, have been proved as follows:

$$e_n \leq 2nK \exp\left(-\frac{n-K}{18H_c}\right) \tag{1}$$

**Algorithm 3** SR algorithm. Let $A_1 = \{1, \ldots, K\}$, $\overline{\log}(K) = \frac{1}{2} + \sum_{i=2}^{K} \frac{1}{i}$, $n_0 = 0$ and for $k \in \{1, \ldots, K-1\}$, $n_k = \left\lceil \frac{1}{\overline{\log}(K)} \frac{n-K}{K+1-k} \right\rceil$.

**Require:** $n$ (total number of rounds)
1: **for** $k \leftarrow 1, K-1$ **do** ▷ Exploration loop
2: **for** $i \in A_k$ **do** ▷ Pull arms in $A_k$
3: Pull arm $i$ for $n_k - n_{k-1}$ rounds and get rewards
4: **end for**
5: $A_{k+1} \leftarrow A_k \setminus \arg\min_{i \in A_k} \widehat{X}_{i,n_k}$ ▷ Reject minimal arm in $A_k$
6: **end for**
7: **return** $J_n \in A_k$ ▷ $J_n$ is the unique element of $A_k$

**Algorithm 4** UGapE algorithm. $\beta_k(t-1)$ is a confidence interval, and $U_k(t)$ and $L_k(t)$ are high-probability upper and lower bounds. For arm $k \in A$ and round $t$, $U_k(t) = \widehat{X}_{k,T_k(t-1)} + \beta_k(t-1)$, $L_k(t) = \widehat{X}_{k,T_k(t-1)} - \beta_k(t-1)$ and $\beta_k(t-1) = \sqrt{\frac{a}{T_k(t-1)}}$.

**Require:** $a > 0$ (exploration parameter), $n$ (total number of rounds)
1: **for** $i \leftarrow 1, K$ **do** ▷ Initialization
2: Pull arm $i$ and get reward $X_{i,1}$
3: **end for**
4: **for** $t \leftarrow K+1, n$ **do** ▷ Exploration loop
5: **for** $k \leftarrow 1, K$ **do** ▷ Calculate score $S_k$ of arm $k$
6: $S_k = \max_{i \neq k} U_i(t) - L_k(t)$
7: **end for**
8: $l_t \in \arg\min_{k \in A} S_k$
9: $u_t \in \arg\max_{j \in A \setminus l_t} U_j(t)$ ▷ $u_t$: the best possible arm left outside $l_t$
10: Draw arm $I_t \in \arg\max_{k \in \{l_t, u_t\}} \beta_k(t-1)$ and get reward $X_{I_t, T_{I_t}(t-1)+1}$
11: **end for**
12: **return** $J_n \in \arg\max_{i \in A} \widehat{X}_{i,T_i(n)}$

for UCB-E and

$$e_n \leq 2nK \exp\left(-\frac{n-K}{2H_a}\right) \tag{2}$$

for UGapE where

$$c = \frac{25}{36}\frac{n-K}{H_c} \text{ and } a = \frac{n-K}{4H_a}. \tag{3}$$

The variables $H_c$ and $H_a$, called complexities of a problem, are calculated by using the reward distribution $\nu_1, \nu_2, \ldots, \nu_K$. However, such distribution is unknown in advance. As automatic parameter tuning methods (UCB-E auto and UGapE auto), we calculate suitable estimates $\widehat{H}_c$ and $\widehat{H}_a$ of $H_c$ and $H_a$ from observations at each round $t$ and select an arm using the estimates. We show the details of these methods in Supplementary Methods.

### 2.2 Preparation of docking poses

It is necessary to prepare the correct (stable) binding pose of a protein and a ligand to calculate the binding free energy between them.

**Table 1.** Basic information of the eight complexes in the test set

| Protein | PDB code | Ligand ID | Molecular weight | No. of poses | Correct poses | Average RMSD (Å) |
|---|---|---|---|---|---|---|
| CDK2 | 2R3J | SCJ | 380.2 | 20 | 1 | 7.81 |
| CDK2 | 1OI9 | N20 | 339.4 | 20 | 1 | 5.67 |
| CDK2 | 1KE6 | LS2 | 401.5 | 20 | 3 | 6.49 |
| CDK2 | 3DDQ | RRC | 354.5 | 20 | 3 | 7.14 |
| FA10 | 2WYG | 461 | 472.0 | 20 | 3 | 4.98 |
| FA10 | 2J94 | G15 | 489.0 | 20 | 2 | 5.66 |
| HS90A | 2VCI | 2GJ | 465.5 | 20 | 1 | 6.61 |
| HS90A | 3VHD | VHE | 385.5 | 20 | 1 | 6.25 |

In this paper, we consider that a pose is correct if the root mean square deviation (RMSD) of the binding pose is <2.0 Å from the experimentally observed conformation. This criteria is widely used in binding pose prediction (Cheng *et al.*, 2009; Hou *et al.*, 2011; Thompson *et al.*, 2008).

We prepared 20 docking poses for each protein–ligand complex, which consists of three proteins, CDK2, HSP20A and FA10 and eight ligands (see Supplementary Methods for details and Supplementary Material). The basic information of the dataset is listed in Table 1. We show the RMSDs and the docking scores of the prepared docking poses for each of all the compounds in Supplementary Tables S1 and S2. In our dataset, one to three correct poses (<2.0 Å) are included in each pose set.

## 2.3 MD simulations of protein–ligand complexes

In molecular mechanics (MM) minimization and MD simulations, the Amber99SB-ILDN force field (Lindorff-Larsen *et al.*, 2010) was used for proteins and the general AMBER force field (Wang *et al.*, 2004) was used for ligands. The TIP3P water model (Jorgensen *et al.*, 1983) was used for water molecules. Water molecules were placed around the complex model with an encompassing distance of 8 Å, including roughly 13 000 water molecules. Charge-neutralization ions were added to neutralize the system.

All MD simulations were carried out in periodic boundary conditions using the GROMACS 4 program (Hess *et al.*, 2008) on the K-computer (RIKEN, Japan). Energy was first minimized for the initial configuration using the steepest descent method, then the Particle Mesh Ewald method (Darden *et al.*, 1993) was used to calculate the long-range electrostatic interactions. Hydrogen atoms are constrained using the LINCS (Hess, 2008) algorithm. After minimization, the system was equilibrated for 100 ps under constant volume (NVT) and run for 100 ps under constant pressure and temperature (NPT) with positional restraints on protein heavy atoms and ligand atoms. Initial velocities were assigned from a Maxwell distribution at 298 K. Then a 1 ns production run was conducted under the NPT condition without positional restraints. In this procedure, the temperature was maintained at 298 K and the pressure was maintained at 1 atm.

Twenty sets of 1 ns production runs were performed with different initial velocities for each docking pose. The total simulation time was 3.2 $\mu$s (1 ns × 20 runs with different velocities × 20 poses × 8 complexes). All MD runs were carried out with time steps of 2 fs and snapshots for the MM-PBSA analysis were taken every 10 ps.

## 2.4 MM-PBSA calculations

In the MM-PBSA method, the total free energy $G$ of a biomolecular system is expressed as follows (Kollman *et al.*, 2000):

$$G = E_{\mathrm{MM}} + G_{\mathrm{PB}} + G_{\mathrm{SA}} - TS_{\mathrm{solute}} \quad (4)$$

$$E_{\mathrm{MM}} = E_{\mathrm{int}} + E_{\mathrm{el}} + E_{\mathrm{vdw}} \quad (5)$$

where $E_{\mathrm{MM}}$ is MM energy term consisting of internal $E_{\mathrm{int}}$ (from bonds, angles and dihedral angles), non-bonded electrostatic $E_{\mathrm{el}}$ and van der Waals $E_{\mathrm{vdw}}$ energies. The polar and non-polar contributions to the solvation free energies are expressed as a continuum solvent Poisson–Boltzmann (PB) model and a solvent-accessible surface area (SASA)-dependent non-polar solvation term, respectively. The last term in (4) is the absolute temperature $T$ multiplied by the entropy $S$. The binding free energy $\Delta G_{\mathrm{bind}}$ is expressed as follows:

$$\Delta G_{\mathrm{bind}} = G_{\mathrm{complex}} - \left(G_{\mathrm{protein}} + G_{\mathrm{ligand}}\right). \quad (6)$$

To calculate the binding free energy, the typical procedure is to average $\Delta G_{\mathrm{bind}}$ in a set of conformation ensembles of the given complex structure taken from MD simulations (Hou *et al.*, 2010; Kollman *et al.*, 2000). In this study, conformation ensembles in 1 ns MD production were used, as it was suggested that MM-PBSA calculation based on short (∼1 ns) MD simulations is appropriate for $\Delta G_{\mathrm{bind}}$ prediction and pose rescoring (Hou *et al.*, 2010, 2011; Xu *et al.*, 2013).

MM-PBSA calculations were carried out using the MMPBSA.py module (Miller III *et al.*, 2012) in the Amber12 package (Case *et al.*, 2012). In the calculation, we used the single-trajectory protocol, which is much faster than the original separate-trajectory protocol (Hou and Yu, 2007).

The snapshots for MM-PBSA calculations were taken every 10 ps over the last 500 ps period in each MD production run, resulting in a total of 50 snapshots per MD run. The $\Delta G_{\mathrm{bind}}$ estimate for each MD run was calculated by averaging the 50 MM-PBSA estimates for the snapshots. The non-polar solvation free energy ($G_{\mathrm{SA}}$) was determined by the SASA using equation:

$$G_{\mathrm{SA}} = \gamma \mathrm{SASA} + \beta \quad (7)$$

where the surface tension $\gamma$ and the offset $\beta$ were set to the standard values of 0.00542 kcal mol$^{-1}$ Å$^{-2}$ and 0.92 kcal/mol, respectively. It has been suggested many times that the change of conformation entropy $TS_{\mathrm{solute}}$ term can be omitted in pose prediction as it does not always improve the prediction results (Hou *et al.*, 2010, 2011; Kumari *et al.*, 2014; Yang *et al.*, 2011), and therefore, in light of its high computational cost, this term was not considered here.

## 2.5 Calculated $\Delta G_{\mathrm{bind}}$ distributions of the prepared poses

For the generated conformation ensembles in the MD productions, we calculated $\Delta G_{\mathrm{bind}}$s according to the above MM-PBSA calculation procedure. For each pose, we calculated the average ($\overline{\Delta G_{\mathrm{20vel.}}}$) and SD of the 20 $\Delta G_{\mathrm{bind}}$. Calculated $\overline{\Delta G_{\mathrm{20vel.}}}$ and SDs of $\Delta G_{\mathrm{bind}}$ are listed in Supplementary Tables S1 and S2. In our dataset, the $\overline{\Delta G_{\mathrm{20vel.}}}$ identified correct binding poses in each complex, whereas

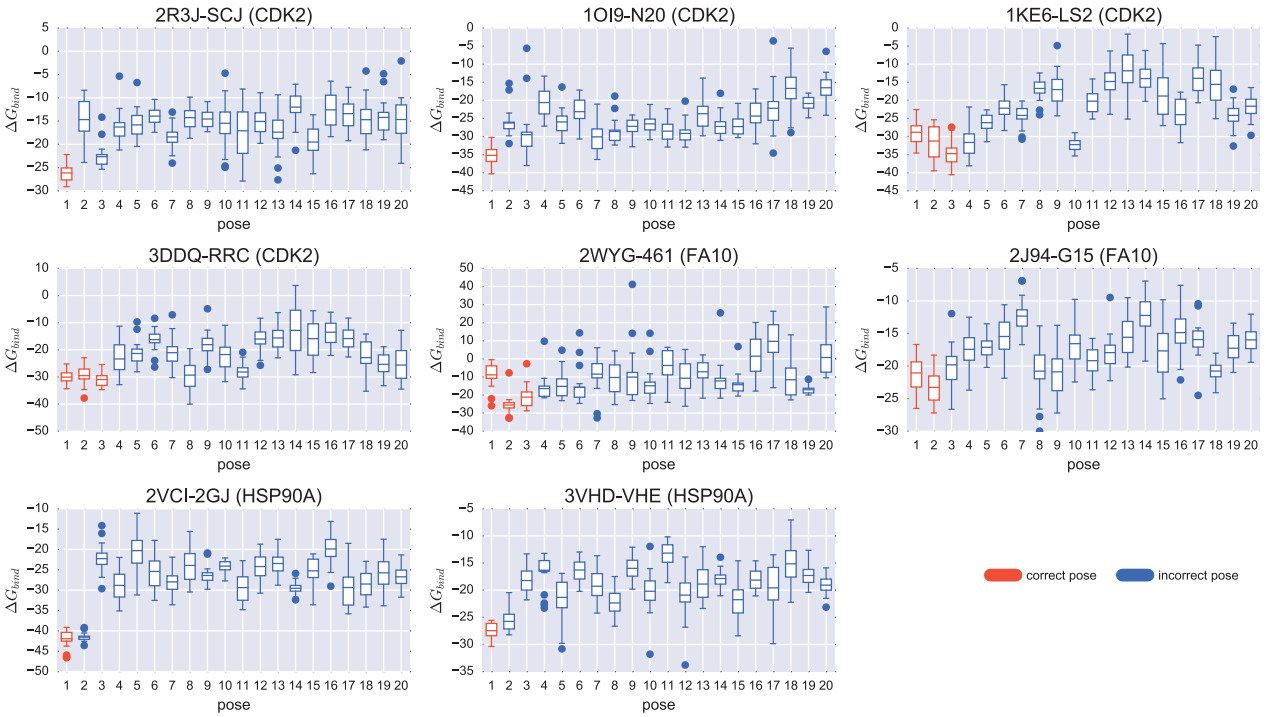


**Fig. 4.** The distributions of calculated $\Delta G_{\text{bind}}$s for all poses in eight compounds. Twenty $\Delta G_{\text{bind}}$ values are calculated for each pose. Red poses are the correct binding poses (RMSD < 2.0 Å) and blue ones are incorrect. Horizontal lines represent within boxes the mean values and indicate the mean value and the first and last quartile, while the ends of the whiskers show maximum and minimum values within 1.5 IQR (inter-quartile range: the distance between the first and last quartiles) of the first and last quartiles (Color version of this figure is available at *Bioinformatics* online.)

only two correct poses (2WYG-461 and 2J94-G15) were identified by the docking score. Figure 4 shows the distributions of calculated $\Delta G_{\text{bind}}$s for all binding poses for eight compounds. The $\Delta G_{\text{bind}}$ values of correct binding poses (red) are comparatively smaller than those of incorrect poses (blue). Note that some $\Delta G_{\text{bind}}$ values for incorrect poses are smaller than for correct poses.

We show the details of tool information and their parameters for the docking pose preparations, MD simulations and MM-PBSA calculations in Supplementary Table S3.

## 3 Results and discussion

To show the effectiveness of the proposed method, we performed a pose prediction experiment on our dataset. In the experiment, a pose prediction trial is a process to choose a pose from 20 pose candidates by calculating $\Delta G_{\text{bind}}$ values with MD and MM-PBSA for each pose. In the pose prediction trials, the number of MD and MM-PBSA runs of each pose is controlled by different allocation algorithms: uniform sampling (baseline method), UCB($p$), UCB-E (auto), SR and UGapE (auto) (see Section 2 for more details). Note that a pose prediction trial may succeed, i.e. the pose chosen by the trial is a correct pose, or fail while using the same algorithm due to variations of $\Delta G_{\text{bind}}$ depending on the initial velocity distribution in MD, as shown in Figure 4.

Figure 5 illustrates the reductions of MD and MM-PBSA runs in a pose prediction trial for 3DDQ-RRC and 2VCI-2GJ. The reduction results of the other compounds are shown in Supplementary Figure S1. Here, we first performed a pose prediction trial using uniform sampling ($k = 10$). Green bars show the number of runs per pose. The total number of runs was 200 ($10 \times 20$ poses). Blue, purple, red and orange bars show the number of runs per pose using UGapE auto, UCB-E auto, SR and UCB($p$) ($p = 4$). Total runs of UGapE auto, UCB-E auto and UCB($p$) were 50, and those of SR was 75. Black lines are the averaged binding free energies ($\overline{\Delta G_{20\text{vel.}}}$) in Supplementary Tables S1 and S2. As shown in Figure 5, the computational resources for the BAI algorithms are concentrated on small $\overline{\Delta G_{20\text{vel.}}}$ poses. And the total numbers of runs using the BAI algorithms are reduced from 200 ($10 \times 20$ poses) runs using uniform sampling to 50 and 75 without reducing the number of runs for promising poses, which have small $\overline{\Delta G_{20\text{vel.}}}$ values. In all the trials in Figure 5, correct poses were successfully chosen. However, such trials do not always succeed due to the large fluctuations of $\Delta G_{\text{bind}}$ (see Fig. 4).

To evaluate the performance of the BAI algorithms, we estimated the probability of correct pose prediction, i.e. the probability that an algorithm selects the correct pose among pose candidates under fixed parameters. For a given allocation algorithm and total number of runs, we repeated pose prediction trials and calculated the ratio of succeeded trials to total trials. Instead of actually performing the MD and MM-PBSA runs, we estimated the probability by sampling without replacement a binding free energy $\Delta G_{\text{bind}}$ from the energies calculated in Sections 2.3–2.5 and shown in Figure 4. (When all the calculated binding free energies $\Delta G_{\text{bind}}$s of a pose were used up, pose selection was done from the remaining poses except for the pose.) We show the result of probability of correct pose prediction using uniform sampling (baseline) for all the complexes in Supplementary Figure S2. Figure 6 shows the probabilities of correct pose prediction using BAI algorithms (UGapE auto, UCB-E auto, SR and UCB($p$) ($p = 4$)) increasing the total number of MD and MM-PBSA runs. For all the algorithms, the probabilities rose with increasing the number of total runs. For all the compounds, the total numbers of MD and MM-PBSA runs using BAI algorithms were reduced without sacrificing the probability of correct pose prediction. We summarized the total number of MD and MM-PBSA runs

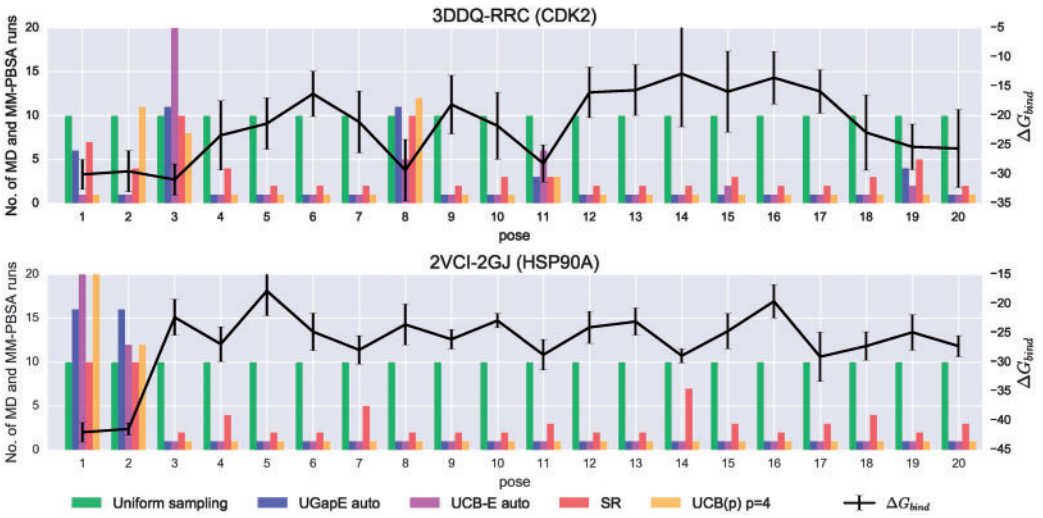


**Fig. 5.** Reductions of MD and MM-PBSA runs per pose by BAI algorithms in a pose prediction trial. Green bars show the numbers of runs ($k = 10$) per pose by uniform sampling in a pose prediction trial. Blue, purple, red and orange bars show the number of runs per pose using UGapE auto, UCB-E auto, SR and UCB($p$) ($p = 4$). Black lines are the averaged binding free energies ($\overline{\Delta G_{20\text{vel.}}}$). The total numbers of runs by BAI algorithms are reduced from 200 ($10 \times 20$ poses) by uniform sampling to 50 and 75 without reducing the number of runs for promising poses, which have small $\overline{\Delta G_{20\text{vel.}}}$ values (Color version of this figure is available at *Bioinformatics* online.)

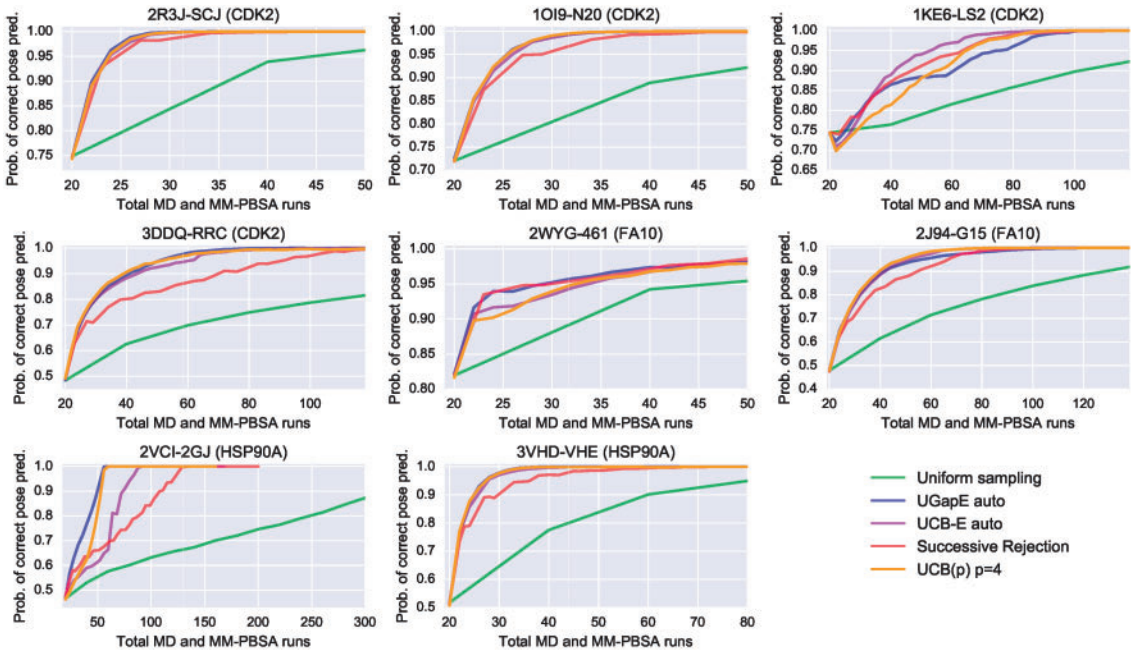


**Fig. 6.** The probabilities of correct pose prediction by the proposed methods and uniform sampling (baseline) at different numbers of MD and MM-PBSA runs for eight complexes. The total numbers of MD and MM-PBSA runs (computational cost) were reduced using the BAI algorithms [UGapE auto, UCB-E auto, UCB($p$) $p = 4$ and SR] without sacrificing accuracy compared to uniform sampling (green). The probabilities of UGapE auto (blue) and UCB-E auto (purple), whose exploration parameters were automatically adjusted, are higher than those of uniform sampling (green). Although UCB($p$) showed almost the same high performance as UGapE auto and UCB-E auto under the exploration parameter $p = 4$, the result varies depending on the parameter as shown in Supplementary Figure S3. The result using SR (red) is a little worse than other BAI algorithms (Color version of this figure is available at *Bioinformatics* online.)

to reach the correct pose prediction probability of 95% for exploration parameter free algorithms in Table 2. Table 3 shows the probabilities of correct pose prediction with the total number of runs fixed to 40, 60 and 100. From Table 2, the average numbers of runs using UGapE auto and UCB-E auto were approximately reduced by a factor of 3.5 compared to uniform sampling. The average probabilities of correct pose prediction using UGapE auto and UCB-E auto reached 90% with only 40 total runs, although the probability of uniform sampling was 76% (Table 3).

In particular, when pose prediction is difficult due to a small difference in $\overline{\Delta G_{20\text{vel.}}}s$ between correct and incorrect poses, such as for 3DDQ-RRC and 2VCI-2GJ, computational cost using UGapE auto can be greatly reduced by a factor of 5.00 and 6.67, respectively (Fig. 6 and Table 2). For example, the difference of $\overline{\Delta G_{20\text{vel.}}}$ values between pose 1 (−41.9 kcal/mol, correct) and pose 2 (−41.6 kcal/mol, incorrect) of 2VCI-2GJ was 0.37 kcal/mol (Supplementary Table S2), whereas the SDs of poses 1 and 2 were relatively large (2.23 and 1.00 kcal/mol). When the difference is small, a large

**Table 2.** The number of MD and MM-PBSA runs to reach the correct pose prediction probability of 95% using exploration parameter free algorithms

| Algorithm | 2R3J-SCJ | 1OI9-N20 | 1KE6-LS2 | 3DDQ-RRC | 2WYG-461 | 2J94-G15 | 2VCI-2GJ | 3VHD-VHE | Average |
|---|---|---|---|---|---|---|---|---|---|
| Uniform sampling | 60 | 60 | 160 | 260 | 60 | 160 | 360 | 100 | 153 |
| UGapE auto | **24 (2.50)** | **26 (2.31)** | 75 (2.13) | **52 (5.00)** | **29 (2.07)** | 57 (2.81) | **54 (6.67)** | **28 (3.57)** | **43.1 (3.54)** |
| UCB-E auto | **24 (2.50)** | **26 (2.31)** | **53 (3.02)** | 60 (4.33) | 34 (1.76) | **51 (3.14)** | 81 (4.44) | **28 (3.57)** | 44.6 (3.42) |
| SR | 27 (2.22) | 29 (2.07) | 65 (2.46) | 92 (2.83) | 34 (1.76) | 68 (2.35) | 124 (2.90) | 38 (2.63) | 59.6 (2.56) |

*Note*: The reduction effects obtained by dividing the numbers of runs of the BAI algorithms by the ones of uniform sampling are shown in parentheses. The bold value indicates the result of the largest reduction effect in each complex.

**Table 3.** The probabilities (%) of correct pose prediction with the total number of runs fixed to 40, 60 and 100

| Algorithm | Total runs | 2R3J-SCJ | 1OI9-N20 | 1KE6-LS2 | 3DDQ-RRC | 2WYG-461 | 2J94-G15 | 2VCI-2GJ | 3VHD-VHE | Average |
|---|---|---|---|---|---|---|---|---|---|---|
| Uniform sampling | 40 | 93.9 | 88.9 | 76.5 | 62.6 | 94.2 | 61.5 | 53.1 | 77.5 | 76.0 |
| UGapE auto | 40 | **100** | **100** | 86.5 | **89.5** | **97.4** | **89.1** | **76.9** | **100** | **92.4** |
| UCB-E auto | 40 | **100** | **100** | **89.0** | 88.3 | 96.7 | 88.6 | 59.1 | 99.7 | 90.2 |
| SR | 40 | 99.9 | 99.3 | 86.2 | 79.9 | 96.6 | 81.9 | 63.6 | 96.9 | 88.0 |
| Uniform sampling | 60 | 98.7 | 95.5 | 81.6 | 69.9 | 96.6 | 71.5 | 57.7 | 90.1 | 82.7 |
| UGapE auto | 60 | **100** | **100** | 89.7 | **98.0** | 98.6 | 95.7 | **100** | **100** | **97.7** |
| UCB-E auto | 60 | **100** | **100** | **96.9** | 95.0 | **99.6** | **97.5** | 66.5 | **100** | 94.4 |
| SR | 60 | **100** | **100** | 93.4 | 85.5 | 98.8 | 90.8 | 67.6 | 99.4 | 91.9 |
| Uniform sampling | 100 | **100** | 99.5 | 89.7 | 78.7 | 99.9 | 83.8 | 63.2 | 97.2 | 89.0 |
| UGapE auto | 100 | **100** | 99.8 | **100** | 99.8 | 99.5 | 99.4 | **100** | **100** | 99.8 |
| UCB-E auto | 100 | **100** | **100** | **100** | **99.9** | **100** | **100** | **100** | **100** | **100** |
| SR | 100 | **100** | **100** | 99.9 | 96.7 | **100** | 99.9 | 84.1 | **100** | 97.6 |

The bold value indicates the highest probability in the same total runs.

number of runs for promising poses (e.g. poses 1 and 2 in 2VCI-2GJ) is needed to explore the difference and predict a correct pose, leading to unnecessary calculation of many unpromising poses (e.g. poses 3 to 20 in 2VCI-2GJ) under uniform sampling. On the other hand, BAI algorithms avoid such unpromising poses at an early stage and computational resource can be concentrated on only promising poses, leading to an even greater reduction in computational cost.

In order to assess the robustness of these algorithms, especially for the exploration parameters $p$, $c$ and $a$ for UCB($p$), UCB-E and UGapE algorithms, we calculated the probabilities of correct pose prediction by changing these parameters. Supplementary Figures S3–S5 show the probability curves of correct pose predictions with different parameters. From these results, the optimal value of an exploration parameter for each complex may change depending on its $\Delta G_{bind}$ distributions shown in Figure 4. In Supplementary Figures S3 and S4, the automatic parameter adjustment methods UCB-E auto and UGapE auto are not necessarily the best result for all complexes. However, these exploration parameter free algorithms are practical because it is difficult to decide the optimal parameter among different complexes and takes cost to search it by calculating the probabilities using different parameters as shown in Supplementary Figures S3–S5. Together with the results in Figure 6, it is suggested that UGapE auto and UCBE auto algorithms are promising compared to uniform sampling and the other BAI algorithms.

We believe that our framework based on BAI algorithms can be widely used in various situations. The pose prediction results indicate the effectiveness of our framework even when multiple correct poses are present among pose candidates. In the experiment, the number of binding poses is set to 20 and the number of correct poses is set to 1, 2 or 3, but the proposed method is effective for any number. MD and MM-PBSA calculations were performed as described in Section 2, excluding the entropy term, but even with this entropy term, or with different parameters and methods of calculation, the proposed method would still be applicable. Our framework can be used for such binding pose prediction procedures.

## 4 Conclusions

In this study, we have proposed an efficient binding pose prediction method based on BAI algorithms to estimate binding free energy between ligands and proteins. Our results on our test datasets showed that the proposed method reduced the computational cost and improved accuracies compared to uniform sampling, in particular for small differences in binding free energies between correct and incorrect poses.

While we did confirm the effectiveness of BAI algorithms with the MM-PBSA method, we believe they could be just as effective at controlling all kinds of molecular simulations and find the best results under limited computational resources. In future work, we plan to improve the accuracy of correct pose prediction and binding free energy estimation by considering non-uniform weighting across MD and MM-PBSA runs. The distribution of binding modes and binding free energies of a ligand and protein complex can be effectively estimated by using such controlling algorithms. We would also like to investigate methods using Bayesian optimization (Shahriari *et al.*, 2016) in order to search for optimal poses from many candidates without limiting their number.

## Acknowledgements

We thank T. Shimada and K. Kumazawa (Teijin Pharma Ltd, Tokyo, Japan) and K. Hattori and K. Yasuo (Shionogi & Co., Ltd., Osaka, Japan) for providing the compound docking poses toward HS90A and FA10 proteins using GLIDE (Schrödinger, LLC, NY). We also thank Dr. K. Yoshizoe (RIKEN AIP center, Unit Leader) for helpful comments and suggestions.

## Funding

This research was supported by MEXT as 'Priority Issue on Post-K computer' (Building Innovative Drug Discovery Infrastructure Through Functional Control of Biomolecular Systems). This research used computational resources on the K computer provided by the RIKEN Advanced Institute for Computational Science through the HPCI Research Project (Project ID: hp160213 and hp150272), Innovative Drug Discovery Infrastructure through Functional Control of Biomolecular Systems and Subsidy for Kobe Biomedical Innovation Cluster promoting projects. This work was supported by Core Research for Evolutional Science and Technology (CREST), JST [grant number JPMJCR1502 to K.T. and JPMJCR1311 to Y.O. and H.I.].

*Conflict of Interest*: none declared.

## References

Agrawal,S. and Goyal,N. (2012) Analysis of Thompson sampling for the multi-armed bandit problem. In: *Conference on Learning Theory*, pp. 39.1–39.26.

Åqvist,J. *et al*. (1994) A new method for predicting binding affinity in computer-aided drug design. *Protein Eng*., **7**, 385–391.

Audibert,J.-Y. and Bubeck,S. (2010) Best arm identification in multi-armed bandits. In: *Conference on Learning Theory*, p. 13. Haifa, Israel.

Auer,P. *et al*. (2002) Finite-time analysis of the multiarmed bandit problem. *Mach. Learn*., **47**, 235–256.

Berhanu,W.M. and Hansmann,U.H. (2013) The stability of cylindrin $\beta$-barrel amyloid oligomer models—a molecular dynamics study. *Proteins Struct. Funct. Bioinf*., **81**, 1542–1555.

Berman,H.M. *et al*. (2000) The protein data bank. *Nucleic Acids Res*., **28**, 235–242.

Bubeck,S. *et al*. (2009) Pure exploration in multi-armed bandits problems. In: *International Conference on Algorithmic Learning Theory*, pp. 23–37. Springer.

Case,D. *et al*. (2012) *Amber 12*. University of California, San Francisco, CA.

Cheng,T. *et al*. (2009) Comparative assessment of scoring functions on a diverse test set. *J. Chem. Inf. Model*., **49**, 1079–1093.

Colizzi,F. *et al*. (2010) Single-molecule pulling simulations can discern active from inactive enzyme inhibitors. *J. Am. Chem. Soc*., **132**, 7361–7371.

Coulom,R. (2006) Efficient selectivity and backup operators in Monte-Carlo tree search. In: *International Conference on Computers and Games*, pp. 72–83. Springer.

Darden,T. *et al*. (1993) Particle mesh Ewald: an N log (N) method for Ewald sums in large systems. *J. Chem. Phys*., **98**, 10089–10092.

Fujitani,H. *et al*. (2009) Massively parallel computation of absolute binding free energy with well-equilibrated states. *Phys. Rev. E*, **79**, 021914.

Gabillon,V. *et al*. (2011) Multi-bandit best arm identification. In: *Advances in Neural Information Processing Systems*, pp. 2222–2230.

Gabillon,V. *et al*. (2012) Best arm identification: a unified approach to fixed budget and fixed confidence. In: *Advances in Neural Information Processing Systems*, pp. 3212–3220.

Genheden,S. and Ryde,U. (2010) How to obtain statistically converged MM/GBSA results. *J. Comput. Chem*., **31**, 837–846.

Hess,B. (2008) P-LINCS: a parallel linear constraint solver for molecular simulation. *J. Chem. Theory Comput*., **4**, 116–122.

Hess,B. *et al*. (2008) GROMACS 4: algorithms for highly efficient, load-balanced, and scalable molecular simulation. *J. Chem. Theory Comput*., **4**, 435–447.

Hou,T. and Yu,R. (2007) Molecular dynamics and free energy studies on the wild-type and double mutant HIV-1 protease complexed with amprenavir and two amprenavir-related inhibitors: mechanism for binding and drug resistance. *J. Med. Chem*., **50**, 1177–1188.

Hou,T. *et al*. (2010) Assessing the performance of the MM/PBSA and MM/GBSA methods. 1. The accuracy of binding free energy calculations based on molecular dynamics simulations. *J. Chem. Inf. Model*., **51**, 69–82.

Hou,T. *et al*. (2011) Assessing the performance of the molecular mechanics/Poisson Boltzmann surface area and molecular mechanics/generalized born surface area methods. II. The accuracy of ranking poses generated from docking. *J. Comput. Chem*., **32**, 866–877.

Jorgensen,W.L. *et al*. (1983) Comparison of simple potential functions for simulating liquid water. *J. Chem. Phys*., **79**, 926–935.

Kaufmann,E. and Kalyanakrishnan,S. (2013) Information complexity in bandit subset selection. In: *Conference on Learning Theory*, pp. 228–251.

Kollman,P.A. *et al*. (2000) Calculating structures and free energies of complex molecules: combining molecular mechanics and continuum models. *Acc. Chem. Res*., **33**, 889–897.

Kumari,R. *et al*. (2014) g_mmpbsa—a GROMACS tool for high-throughput MM-PBSA calculations. *J. Chem. Inf. Model*., **54**, 1951–1962.

Lavecchia,A. and Di Giovanni,C. (2013) Virtual screening strategies in drug discovery: a critical review. *Curr. Med. Chem*., **20**, 2839–2860.

Li,L. *et al*. (2010) A contextual-bandit approach to personalized news article recommendation. In: *International Conference on World Wide Web*, pp. 661–670. ACM.

Lindorff-Larsen,K. *et al*. (2010) Improved side-chain torsion potentials for the Amber ff99SB protein force field. *Proteins Struct. Funct. Bioinf*., **78**, 1950–1958.

Lionta,E. *et al*. (2014) Structure-based virtual screening for drug discovery: principles, applications and recent advances. *Curr. Top. Med. Chem*., **14**, 1923–1938.

Mikulskis,P. *et al*. (2012) Binding affinities in the SAMPL3 trypsin and host–guest blind tests estimated with the MM/PBSA and LIE methods. *J. Comput. Aided Mol. Des*., **26**, 527–541.

Miller,B.R.,III *et al*. (2012) MMPBSA.py: an efficient program for end-state free energy calculations. *J. Chem. Theory Comput*., **8**, 3314–3321.

Okimoto,N. *et al*. (2009) High-performance drug discovery: computational screening by combining docking and molecular dynamics simulations. *PLoS Comput. Biol*., **5**, e1000528.

Onufriev,A. *et al*. (2000) Modification of the generalized born model suitable for macromolecules. *J. Phys. Chem. B*, **104**, 3712–3720.

Proctor,E.A. *et al*. (2012) Discrete molecular dynamics distinguishes nativelike binding poses from decoys in difficult targets. *Biophys. J*., **102**, 144–151.

Robbins,H. (1952) Some aspects of the sequential design of experiments. *Bull. Amer. Math. Soc*., **58**, 527–535.

Sadiq,S.K. *et al*. (2010) Accurate ensemble molecular dynamics binding free energy ranking of multidrug-resistant HIV-1 proteases. *J. Chem. Inf. Model*., **50**, 890–905.

Shahriari,B. *et al*. (2016) Taking the human out of the loop: a review of Bayesian optimization. *Proc. IEEE*, **104**, 148–175.

Silver,D. *et al*. (2016) Mastering the game of go with deep neural networks and tree search. *Nature*, **529**, 484–489.

Srinivasan,J. *et al*. (1998) Continuum solvent studies of the stability of DNA, RNA, and phosphoramidate-DNA helices. *J. Am. Chem. Soc*., **120**, 9401–9409.

Thompson,D.C. *et al*. (2008) Investigation of MM-PBSA rescoring of docking poses. *J. Chem. Inf. Model*., **48**, 1081–1091.

Villar,S.S. *et al*. (2015) Multi-armed bandit models for the optimal design of clinical trials: benefits and challenges. *Stat. Sci*., **30**, 199.

Wang,J. *et al*. (2004) Development and testing of a general amber force field. *J. Comput. Chem*., **25**, 1157–1174.

Xu,L. *et al*. (2013) Assessing the performance of MM/PBSA and MM/GBSA methods. 3. The impact of force fields and ligand charge models. *J. Phys. Chem. B*, **117**, 8408–8421.

Yang,T. *et al*. (2011) Virtual screening using molecular simulations. *Proteins Struct. Funct. Bioinf*., **79**, 1940–1951.