The Cloning Method for Counting and Optimization

Technion Excellence Program

Reuven Rubinstein

Faculty of Industrial Engineering and Management, Technion, Israel
Contents

1. Introduction: Approximate Counting.
5. The Splitting Method as a Natural Extension of the Classic Randomized Algorithms.
6. Why Classic Randomized Algorithms do not Work?
7. Why Non-Classic Splitting Algorithms do Work?
8. Applications: Integer Programs, the Satisfiability Problem, Vertex Coloring, etc.
9. Convergence and Numerical Results.
10. Open Problems.
Counting Hamiltonian Cycles

How many Hamiltonian cycles does this graph have?
Vertex Coloring

Given a graph $G = (V, E)$ with $m$ edges and $n$ vertices, color the vertices of $V$ with given $q$ colors (say 2 colors), such that for each edge $(i, j) \in E$, vertices $i$ and $j$ have different colors.

Given $q$ colors, how many different Vertex Coloring this graph has?
Consider a graph $G = (V, E)$ with $m$ edges and $n$ vertices. A node set is called *independent* if no two nodes are connected by an edge, that is, if no two nodes are adjacent; see Figure 1 for an illustration of this concept.

Figure 1: The black nodes form an independent set since they are not adjacent to each other.
Consider a set containing both equality and inequality constraints of an integer program, that is

\[ \sum_{k=1}^{n} a_{ik} x_k = b_i, \ i = 1, \ldots, m_1, \]

\[ \sum_{k=1}^{n} a_{jk} x_k \geq b_j, \ j = m_1 + 1, \ldots, m_1 + m_2, \]

\[ \mathbf{x} \geq \mathbf{0}, \ x_k \text{ integer } \forall k = 1, \ldots, n. \]
Counting via Monte Carlo

We start with the following basic **Example**.
Assume we want to calculate an area of same “irregular" region \( \mathcal{X}^* \). The Monte-Carlo method suggests inserting the ”irregular" region \( \mathcal{X}^* \) into a nice “regular" one \( \mathcal{X} \) as per figure below:

\[ \mathcal{X} : \text{Set of objects (paths in a graph, colorings of a graph, etc.)} \]
\[ \mathcal{X}^* : \text{Subset of special objects (cycles in a graph, colorings of a certain type, etc).} \]
Counting via Monte Carlo

To calculate $|\mathcal{X}^*|$ we apply the following sampling procedure:

(i) Generate a random sample $X_1, \ldots, X_N$, uniformly distributed over the “regular” region $\mathcal{X}$.

(ii) Estimate the desired area $|\mathcal{X}^*|$ as

$$|\hat{\mathcal{X}}^*| = \hat{\ell}|\mathcal{X}|,$$

where

$$\hat{\ell} = \frac{N_{\mathcal{X}^*}}{N_{\mathcal{X}}} = \frac{1}{N} \sum_{k=1}^{N} I\{x_k \in \mathcal{X}^*\},$$

$I\{x_k \in \mathcal{X}^*\}$ denotes the indicator of the event $\{X_k \in \mathcal{X}^*\}$ and $\{X_k\}$ is a sample from $f(x)$ over $\mathcal{X}$, where $f(x) = \frac{1}{|\mathcal{X}|}$. 
The Randomized Algorithm

Estimating $|\mathcal{X}^*|$ with a known $|\mathcal{X}_0| = |\mathcal{X}|$.

1. Define a sequence of sets $\mathcal{X}_1, \ldots, \mathcal{X}_m$ and write $|\mathcal{X}^*|$ as

$$|\mathcal{X}^*| = |\mathcal{X}_0| \prod_{t=1}^{m} \frac{|\mathcal{X}_t|}{|\mathcal{X}_{t-1}|},$$

Note that the ratio $\frac{|\mathcal{X}^*|}{|\mathcal{X}_0|}$ is very small, like $= 10^{-100}$, while each ratio $c_t = \frac{|\mathcal{X}_t|}{|\mathcal{X}_{t-1}|}$ is not, like $c_t = 10^{-2}$ or greater and $\mathcal{X}_0 \supset \mathcal{X}_1 \supset \cdots \supset \mathcal{X}_m = \mathcal{X}^*$.

2. Develop an efficient estimator for each $c_t$ and deliver

$$|\hat{\mathcal{X}}^*| = |\mathcal{X}_0| \prod_{t=1}^{m} \hat{c}_t.$$
Consider an arbitrary ordering of the edges. Let $E_j$ be the set of the first $j$ edges and let $G_j = (V, E_j) = (V, \{e_1, \ldots, e_j\})$ be the associated sub-graph. Note that $G_m = G$, and that $G_{j+1}$ is obtained from $G_j$ by adding the edge $e_{j+1}$, which is not in $G_j$. Denoting $\mathcal{X}_j$ the set of independent sets of $G_i$ we can write $|\mathcal{X}^*| = |\mathcal{X}_m|$ in the form $|\mathcal{X}^*| = |\mathcal{X}_0| \prod_{t=1}^{m} \frac{|\mathcal{X}_t|}{|\mathcal{X}_{t-1}|}$. Here $|\mathcal{X}_0| = 2^n$, since $G_0$ has no edges and thus every subset of $V$ is an independent set, including the empty set. Note that here $\mathcal{X}_0 \supset \mathcal{X}_1 \supset \cdots \supset \mathcal{X}_m = \mathcal{X}^*$. 
Vertex Coloring

Given a graph $G = (V, E)$ with $m$ edges and $n$ vertices, color the vertices of $V$ with given $q$ colors, such that for each edge $(i, j) \in E$, vertices $i$ and $j$ have different colors. Here, as before, we consider an arbitrary ordering of the edges. Let $E_j$ be the set of the first $j$ edges and let $G_j = (V, E_j)$ be the associated sub-graph. Note that $G_m = G$, and that $G_{j+1}$ is obtained from $G_j$ by adding the edge $e_{j+1}$. Here $|\mathcal{X}_0| = q^n$, since $G_0$ has no edges.
The Rare-Event Approach

It is often more convenient to cast the problem of estimating \(|\mathcal{X}^*|\) into the problem of estimating the rare event probability

\[ \ell(m) = \frac{|\mathcal{X}^*|}{|\mathcal{X}|}, \]

which can be also written as

\[ \ell(m) = \mathbb{E}_f \left[ I\{S(X) \geq m\} \right]. \]

Here \(S(X)\) is the sample performance, like the length of a randomly selected Hamiltonian cycle, \(X \sim f(x)\), \(f(x)\) is typically being uniformly distributed on the set of points of \(\mathcal{X}\) and \(m\) is as before, a fixed parameter.
We estimate $\ell(m)$ as

$$\ell(m) = c_0 \prod_{t=1}^{T} c_t, = \mathbb{E}_f \left[ I_{\{S(x) \geq m\}} \right]$$

where, as before $c_0 = \mathbb{E}_f[I_{\{S(x) \geq m_0\}}]$, 

$$c_t = |\mathcal{X}_t|/|\mathcal{X}_{t-1}| = \mathbb{E}_{g_{t-1}^*} [I_{\{S(x) \geq m_t\}}].$$

$\{m_t, t = 0, 1, \ldots, T\}$ is a fixed grid satisfying $-\infty < m_0 < m_1 < \ldots m_T = m$ (typically $m_t = t$). Here $g_{t-1}^* = \mathcal{U}(\mathcal{X}_{t-1})$ and $\mathcal{X}_t = \{x : S(x) \geq m_t\}$.  

\[\]

The Cloning Method for Counting and Optimization  \textit{Technion Excellence Program}  \textendash{} p. 13/4
The Rare-Event Approach

The final estimator of $\ell(m)$, based on the product of

c_t = \mathbb{E}_{g^*_{t-1}}[I\{S(x) \geq m_t\}], \ t = 0, \ldots, T \text{ can be written as}

$$\hat{\ell}(m) = \prod_{t=1}^{T} \hat{c}_t = \frac{1}{NT+1} \prod_{t=0}^{T} N_t,$$

where

$$\hat{c}_t = \frac{1}{N} \sum_{i=1}^{N} I\{S(x_i) \geq m_t\} = \frac{N_t}{N},$$

$$N_t = \sum_{i=1}^{N} I\{S(x_i) \geq m_t\}, \ X_i \sim g^*_{t-1} \text{ and } g^*_{t-1} = f.$$

*The main trick is to show how to sample uniformly from the IS pdf $g^*(x, m_{t-1})$ in the reduced space $\mathcal{X}_t$. 
To sample uniformly in the reduced space $\mathcal{X}_t$ we shall use a combination of the Gibbs sampler with classic splitting method. It is based on running multiple trajectories in parallel. The algorithm will be called the splitting or cloning algorithm.

Our splitting algorithm generates an adaptive sequence of non-parametric tuples

$$
\{(m_0, f(x, v_0)), (m_1, g^*(x, m_0)), \ldots, (m_T, g^*(x, m_{T-1}))\}
$$
Points denoted as ⋆ are uniformly distributed on the sets $\mathcal{X}_0$ and $\mathcal{X}_1$.
Points denoted as • are approximately uniformly distributed on the sets $\mathcal{X}_1$ and $\mathcal{X}_2$. 
General Case: Integer Constraints

Consider a set containing both equality and inequality constraints of an integer program, that is

\[
\sum_{k=1}^{n} a_{ik} x_k = b_i, \quad i = 1, \ldots, m_1, \\
\sum_{k=1}^{n} a_{jk} x_k \geq b_j, \quad j = m_1 + 1, \ldots, m_1 + m_2, \\
\mathbf{x} \geq \mathbf{0}, \quad x_k \text{ integer } \forall k = 1, \ldots, n.
\]
General Case: Integer Constraints

It can be shown that in order to count the number of points (feasible solutions) of the above set one can consider the following associated rare-event probability problem

$$\ell(m) = \mathbb{E}_u \left[ I\{\sum_{i=1}^{m} C_i(X) \geq m\} \right],$$

where the first $m_1$ terms $C_i(X)$’s are

$$C_i(X) = I\{\sum_{k=1}^{n} a_{ik}X_k = b_i\}, \ i = 1, \ldots, m_1,$$

while the remaining $m_2$ ones are

$$C_i(X) = I\{\sum_{k=1}^{n} a_{ik}X_k \geq b_i\}, \ i = m_1 + 1, \ldots, m_1 + m_2.$$
Thus, in order to count the number of feasible solution on the above set we shall consider an associated rare event probability estimation problem involving a sum of dependent Bernoulli random variables. Such representation is crucial for a large set of counting problems.
Figure 2: Iteration 1 of the splitting algorithm for the 6-sided polytop.
Figure 3: The sub-region corresponding to the elite point with $S = 5$. 
Figure 4: The sub-region corresponding to the elite point with $S = 4$. 
Figure 5: Iteration 2 of the splitting algorithm for the 6-sided polytop.
Figure 6: The sub-region corresponding to the elite point with $S = 5$. 
Figure 7: The sub-region corresponding to the elite point with $S = 5$. 
Figure 8: Iteration 3 of the splitting algorithm for the 6-sided polytop.
General Procedure

As mentioned we cast a original counting problem into an associated rare-events probability estimation problem, that estimation of

\[ \ell = \mathbb{P}(S(X) \geq m) = \mathbb{E} \left[ I_{\{S(X) \geq m\}} \right]. \]

and involves the following iterative steps:
1 Starting: Start with the proposal pdf $f(x)$, which is uniformly distributed on the sample space $\mathcal{X}$. Set $t := 1$. 
1 **Starting:** Start with the proposal pdf $f(x)$, which is uniformly distributed on the sample space $\mathcal{X}$. Set $t := 1$.

2 **Update $\hat{m}_t$:** Draw $X_1, \ldots, X_N$ from the uniform pdf $g_t = g(x, \hat{m}_t) = \mathcal{U}(\mathcal{X}_t)$. Find the elite sampling based on $\hat{m}_t$, which is the worst performance of the $\rho \times 100\%$ best performances. Estimate $c_t$ as $\hat{c}_t = N_t/N$. 
1 **Starting:** Start with the proposal pdf $f(x)$, which is uniformly distributed on the sample space $\mathcal{X}$. Set $t := 1$.

2 **Update $\hat{m}_t$:** Draw $X_1, \ldots, X_N$ from the uniform pdf $g_t = g(x, \hat{m}_t) = \mathcal{U}(\mathcal{X}_t)$. Find the elite sampling based on $\hat{m}_t$, which is the worst performance of the $\rho \times 100\%$ best performances. Estimate $c_t$ as $\hat{c}_t = N_t / N$.

3 **Split the elite sample and update $g_t = \mathcal{U}(\mathcal{X}_t)$ and $\mathcal{X}_t$:** Deliver $g_{t+1} = \mathcal{U}(\mathcal{X}_{t+1})$ and $\mathcal{X}_{t+1}$ and increase $t$ by 1.
A General Randomized Algorithm

1 **Starting:** Start with the proposal pdf $f(x)$, which is uniformly distributed on the sample space $\mathcal{X}$. Set $t := 1$.

2 **Update $\hat{m}_t$:** Draw $X_1, \ldots, X_N$ from the uniform pdf $g_t = g(x, \hat{m}_t) = U(\mathcal{X}_t)$. Find the elite sampling based on $\hat{m}_t$, which is the worst performance of the $\rho \times 100\%$ best performances. Estimate $c_t$ as $\hat{c}_t = N_t/N$.

3 **Split the elite sample and update $g_t = U(\mathcal{X}_t)$ and $\mathcal{X}_t$:** Deliver $g_{t+1} = U(\mathcal{X}_{t+1})$ and $\mathcal{X}_{t+1}$ and increase $t$ by 1.

4 **Stopping:** If the stopping criterion is met, then stop; otherwise set $t := t + 1$ and reiterate from step 2.
The Gibbs Sampler

We need to sample from any pdf $g^*(x)$ or any other pdf $g(x)$. It is assumed that generating from the conditional pdfs $g(X_i|X_1, \ldots, X_{i-1}, X_{i+1}, \ldots, X_n)$, $i = 1, \ldots, n$ is simple. In Gibbs sampler for any given vector $X = (X_1, \ldots, X_n) \in \mathcal{X}$ one generates a new vector $\tilde{X} = (\tilde{X}_1, \ldots, \tilde{X}_n)$ as:

**Algorithm: The Systematic Gibbs Sampler**

1. Draw $\tilde{X}_1$ from the conditional pdf $g(X_1|X_2, \ldots, X_n)$.
2. Draw $\tilde{X}_i$ from the conditional pdf $g(X_i|\tilde{X}_1, \ldots, \tilde{X}_{i-1}, X_{i+1}, \ldots, X_n)$, $i = 2, \ldots, n - 1$.
3. Draw $\tilde{X}_n$ from the conditional pdf $g(X_n|\tilde{X}_1, \ldots, \tilde{X}_{n-1})$.

After many *burn-in* periods $\tilde{X}$ is distributed $g(x)$. 

The Cloning Method for Counting and Optimization *Technion Excellence Program* – p. 29/4
Consider estimation

\[ \ell(m) = \mathbb{E}_f \left[ I\{\sum_{i=1}^{n} x_i \geq m\} \right]. \]

The Gibbs sampler for generating variables \( X_i, i = 1, \ldots, N \) is

\[ g^*(x_i, m | \mathbf{x}_{-i}) = c_i(m) f_i(x_i) I\{x_i \geq m - \sum_{j \neq i} x_j\}, \]

where \( |\mathbf{x}_{-i} \) denotes conditioning on all random variables but excluding the remaining ones and \( c_i(m) \) is the normalization constant. Sampling a random variable \( \tilde{X}_i \) can be performed as follows. Generate \( Y \sim \text{Ber}(1/2) \). If \( I\{\tilde{Y} \geq m - \sum_{j \neq i} x_j\} \), then set \( \tilde{X}_i = Y \), otherwise set \( \tilde{X}_i = 1 - Y \).
Originally the capture-recapture method was used to estimate the size, say $M$, of unknown population, under the assumption that two independent samples are taken from that population. To see how the CAP-RECAP method works consider an urn model model with a total of $M$ identical balls. Denote by $N_1$ and $N_2$, the sample sizes taken at the first and the second draw. Assume in addition that

1. The second draw take place only after all $N_1$ balls are returned back to the urn.

2. Before returning the $N_1$ balls back we *mark* each of them, say we paint them in a different color.
Denote by $R$ the number of balls from the first draw that also appear at the second one. Clearly that the estimate of $M$, denoted by $\tilde{M}$ is

$$\tilde{M} = \frac{N_1 N_2}{R}.$$

This is so since

$$\frac{N_2}{M} \approx \frac{R}{N_1}.$$
It is well known that a slightly better unbiased estimate of \( M \) is

\[
\hat{M} = \frac{(N_1 + 1)(N_2 + 1)}{(R + 1)} - 1.
\]  

(1)

The corresponding variance is

\[
\text{Var}(\hat{M}) = \frac{(N_1 + 1)(N_2 + 1)(N_1 - R)(N_2 - R)}{(R + 1)(R + 2)(R + 3)}.
\]  

(2)
Application of The CAP-RECAP to counting problems is trivial. We set $|\mathcal{X}^*| = M$ and note that $N_1$ and $N_2$ correspond to the screened out Gibbs samples at the first and second draws, which are performed after our Algorithm reaches the desired level $m$. As an example, assume that in both experiments (draws) we set originally $N = 10,000$ and then we obtained $N_1 = 5,000$, $N_2 = 5,010$ and $R = 10$. The capture-recapture (CAP-RECAP) estimator of $|\mathcal{X}^*|$, denoted by $|\overline{\mathcal{X}}^*|_{cap}$ is therefore

$$|\overline{\mathcal{X}}^*|_{cap} = 2,505,000.$$
Recall that the regular CAP-RECAP method

1. Is implemented at the last iteration $T$ of the splitting algorithm, that is when some particles have already reached the desired set $\mathcal{X}^*$.  

2. It provides reliable estimators of $|\mathcal{X}^*|$ if it is not too large, say $|\mathcal{X}^*| \leq 10^6$.

Although in typical rare events counting problems, like SAT’s $|\mathcal{X}^*|$ is indeed $\leq 10^6$, nevertheless we present below an enhanced CAP-RECAP version, which extends the original CAP-RECAP for 2-3 more orders, that is it provides reliable counting estimators for $10^8 \leq |\mathcal{X}^*| \leq 10^9$. If not stated otherwise we shall have in mind a SAT problem.
As soon as all $m$ clauses $C_1, \ldots, C_m$ of $\mathcal{X}_m$ are reached (by the splitting algorithm) and it occurs that the resulting product estimator $|\hat{\mathcal{X}}_m|$ of $|\mathcal{X}_m|$ is large, say $> 10^6$ proceed as follows:

1. Generate an (approximately uniform) sample $X_1, \ldots, X_{N\mathcal{X}_m}$ at the desired (final) set $\mathcal{X}_m$ by adding simultaneously some arbitrary auxiliary clauses until for some $\tau$ we have that

$$\frac{N_{\mathcal{X}_{m+\tau}}}{N_{\mathcal{X}_m}} \leq \hat{c}_{m+\tau}. \quad (3)$$

Here say $10^{-2} \leq \hat{c}_{m+\tau} \leq 10^{-3}$; and $N_{\mathcal{X}_m}$ and $N_{\mathcal{X}_{m+\tau}}$ present the respective number of points generated at $\mathcal{X}_m$ and accepted at $\mathcal{X}_{m+\tau}$.

2. Estimate $|\mathcal{X}^*| = |\mathcal{X}|$ by
We call $\hat{\mathcal{X}}_m|_{\text{ecap}}$ the enhanced CAP-RECAP estimator. It is essential to bear in mind that

- $\hat{\mathcal{X}}_{m+\tau}|_{\text{cap}}$ is a CAP-RECAP estimator rather than a splitting (product) one.
- $\hat{\mathcal{X}}_m|_{\text{ecap}}$ does not contain the original estimators $\hat{c}_1, \ldots, \hat{c}_T$ generated by the splitting method.
- Since we only need uniformity of the samples at $\mathcal{X}_m$, we can run the splitting method all the way with relatively small small values of $N$ and $\rho$ until it reaches some vicinity of $\mathcal{X}_m$ and then switch to larger $N$ and $\rho$.
- Formula (4) employs only a single $c$ term.
Comparison of the performance of the product estimator $\hat{X}^*$ with its counterpart $\hat{X}^*_{cap}$ for SAT ($75 \times 305$) model.

| Run | Iter. | $|\hat{X}^*|$ | RE of $|\hat{X}^*|$ | $|\hat{X}^*_{cap}|$ | RE of $|\hat{X}^*_{cap}|$ | $N_1$ | $N_2$ | $R$ |
|-----|-------|-------------|------------------|------------------|------------------|------|------|-----|
| 1   | 21    | 2.67E+04    | 1.39E-01         | 3.07E+04         | 4.49E-02         | 23993| 23908| 186 |
| 2   | 21    | 4.10E+04    | 3.22E-01         | 3.27E+04         | 1.57E-02         | 27064| 26945| 223 |
| 3   | 21    | 2.85E+04    | 8.08E-02         | 3.19E+04         | 7.33E-03         | 26638| 26567| 221 |
| 4   | 21    | 2.96E+04    | 4.36E-02         | 3.09E+04         | 3.83E-02         | 23907| 23993| 185 |
| 5   | 21    | 2.87E+04    | 7.29E-02         | 3.29E+04         | 2.41E-02         | 26967| 27120| 222 |
| 6   | 21    | 3.63E+04    | 1.71E-01         | 3.23E+04         | 4.25E-03         | 26838| 26762| 222 |
| 7   | 21    | 2.39E+04    | 2.28E-01         | 3.30E+04         | 2.64E-02         | 26719| 26697| 216 |
| 8   | 21    | 4.10E+04    | 3.22E-01         | 3.29E+04         | 2.32E-02         | 26842| 26878| 219 |
| 9   | 21    | 2.72E+04    | 1.23E-01         | 3.21E+04         | 1.44E-03         | 26645| 26578| 220 |
| 10  | 21    | 2.70E+04    | 1.29E-01         | 3.21E+04         | 1.75E-03         | 26512| 26588| 219 |
|     | Average | 3.10E+04 | 1.63E-01         | 3.21E+04         | 1.87E-02         | 26512| 26588| 219 |
Numerical Results

Dynamics of one of the runs of the enhanced Algorithm for the random 3-SAT with matrix $A = (75 \times 305)$.

| $t$  | $|\tilde{X}^*|$ | $|\tilde{X}_{cap}^*|$ | $N_t$ | $N_t^{(s)}$ | $m_t^*$ | $m_{*t}$ | $\rho_t$ |
|------|-----------------|-------------------|------|-----------|--------|--------|-------|
| 1    | 4.62E+21        | -                 | 1223 | 1223      | 285    | 274    | 0.122 |
| 2    | 6.88E+20        | -                 | 1490 | 1490      | 288    | 279    | 0.149 |
| 4    | 8.62E+18        | -                 | 1146 | 1146      | 292    | 286    | 0.115 |
| 6    | 2.33E+17        | -                 | 1489 | 1489      | 296    | 290    | 0.149 |
| 9    | 1.93E+15        | -                 | 2635 | 2635      | 298    | 294    | 0.264 |
| 16   | 8.34E+09        | -                 | 1155 | 1155      | 304    | 301    | 0.116 |
| 20   | 3.28E+04        | -                 | 156  | 156       | 305    | 305    | 0.016 |
| 21   | 3.38E+04        | 3.21e+004         | 10000| 8484      | 305    | 305    | 1.000 |
Numerical Results

Performance of splitting algorithm for the 3-SAT (122 × 515) model with $N = 25,000$ and $\rho = 0.1$.

| Run | $N_0$ of iterations | $|\hat{\mathcal{X}}^*|$ | RE of $|\hat{\mathcal{X}}^*|$ | CPU |
|-----|---------------------|-----------------|-----------------|-----|
| 1   | 34                  | 1.41E+06        | 8.41E-02        | 212.32 |
| 2   | 34                  | 1.10E+06        | 2.82E-01        | 213.21 |
| 3   | 34                  | 1.68E+06        | 8.94E-02        | 214.05 |
| 4   | 34                  | 1.21E+06        | 2.14E-01        | 215.50 |
| 5   | 34                  | 1.21E+06        | 2.14E-01        | 214.15 |
| 6   | 34                  | 1.47E+06        | 4.51E-02        | 216.05 |
| 7   | 34                  | 1.50E+06        | 2.81E-02        | 252.25 |
| 8   | 34                  | 1.73E+06        | 1.25E-01        | 243.26 |
| 9   | 34                  | 1.21E+06        | 2.10E-01        | 238.63 |
| 10  | 34                  | 1.88E+06        | 2.24E-01        | 224.36 |
| Average | 34            | 1.44E+06        | 1.52E-01        | 224.38 |
Numerical Results

Performance of the enhanced CAP-RECAP estimator $|\hat{X}^*|_{ecap}$ for the (122 × 515) model along with the regular CAP-RECAP one $|\hat{X}^*|_{cap}$ for the (122 × 520) model.

| Run | Iter | $c_{m+\tau}$ | $|\hat{X}^*|_{cap}$ | RE of $|\hat{X}^*|_{cap}$ | $|\hat{X}^*|_{ecap}$ | RE of $|\hat{X}^*|_{ecap}$ | CPU |
|-----|------|--------------|-------------------|-----------------|-----------------|-----------------|-----|
| 1   | 33   | 3.13E-02     | 5.41E+04          | 1.61E-02        | 1.73E+06        | 1.24E-01        | 138.99 |
| 2   | 34   | 3.47E-02     | 5.51E+04          | 3.44E-03        | 1.59E+06        | 3.38E-02        | 154.64 |
| 3   | 34   | 3.55E-02     | 5.52E+04          | 4.91E-03        | 1.55E+06        | 9.68E-03        | 161.78 |
| 4   | 33   | 4.51E-02     | 5.40E+04          | 1.70E-02        | 1.20E+06        | 2.22E-01        | 163.53 |
| 5   | 34   | 3.04E-02     | 5.13E+04          | 6.71E-02        | 1.69E+06        | 9.54E-02        | 143.27 |
| 6   | 34   | 2.99E-02     | 5.41E+04          | 1.49E-02        | 1.81E+06        | 1.76E-01        | 151.10 |
| 7   | 34   | 4.27E-02     | 5.51E+04          | 3.08E-03        | 1.29E+06        | 1.60E-01        | 174.08 |
| 8   | 34   | 3.87E-02     | 5.42E+04          | 1.35E-02        | 1.40E+06        | 8.86E-02        | 143.57 |
| 12  | 33   | 3.27E-02     | 5.42E+04          | 1.36E-02        | 1.66E+06        | 7.69E-02        | 171.78 |
| 11  | 34   | 4.22E-02     | 5.51E+04          | 2.01E-03        | 1.30E+06        | 1.52E-01        | 151.10 |
| Average | 33.7 | 3.63E-02 | 5.42E+04 | 1.56E-02 | 1.52E+06 | 1.14E-01 | 155.70 |

The Cloning Method for Counting and Optimization
Technion Excellence Program
Numerical Results

Performance of the cloning algorithm for 4-coloring graph for \( n = 40 \) nods, \( N = 100,000 \), \( \rho = 0.1 \)

| Run | \( N_0 \) | Iterations | \( |\tilde{X}^*| \) | RE of \( |\tilde{X}^*| \) | \( |\hat{X}^*_{dir}| \) | RE of \( |\hat{X}^*_{dir}| \) | CPU |
|-----|-----------|------------|----------------|----------------|----------------|----------------|-----|
| 1   | 25        | 1415.62    | 0.055          | 1318           | 0.018          | 257.168        |
| 2   | 25        | 1194.38    | 0.110          | 1322           | 0.015          | 257.098        |
| 3   | 25        | 1356.09    | 0.010          | 1316           | 0.019          | 256.942        |
| 4   | 25        | 1596.61    | 0.190          | 1264           | 0.058          | 258.255        |
| 5   | 25        | 1348.93    | 0.005          | 1316           | 0.019          | 256.568        |
| 6   | 25        | 1627.90    | 0.213          | 1328           | 0.010          | 258.743        |
| 7   | 25        | 1353.55    | 0.009          | 1304           | 0.028          | 257.663        |
| 8   | 25        | 1293.32    | 0.036          | 1330           | 0.009          | 260.002        |
| 9   | 25        | 1229.90    | 0.084          | 1312           | 0.022          | 259.695        |
| 10  | 25        | 1590.67    | 0.185          | 1334           | 0.006          | 259.309        |
| Average | 25 | 1400.7    | 0.090          | 1314           | 0.021          | 258.144        |
THANK YOU