The average compute time of $\Pi$ is $(2m + 1)/2 \cdot l^*$. The average case compute time of $\Pi'$ is given by

$$1/2m \cdot l^*/2 \cdot \sum_{k=0}^{2m} (1-l^*/2)^k + (l^*/2)^2/2l \cdot 1/2n \cdot \sum_{k=0}^{2m-1} (1-l^*/2)^k (2m - 1 - k).$$

The two opposite possible situations are: $l^*/l \rightarrow 1$ and $l^*/l \rightarrow 0$. In the best situation $l^*/l \rightarrow 1$ the average compute time is $l^*/2$ and therefore randomization of type (2) reduces the compute time by $2m + 1$ times. In the worst situation $l^*/l \rightarrow 0$, the ratio is 1 and therefore no reduction is obtained. In both cases, the compute time keeps to be linear in $l^*$. We observe that, randomization of type (2) exploits the existence of paths that are not excessively longer than the shortest path.

Now, we focus on non-blind randomization of type (2), providing a negative result.

**Lemma 4.** Any algorithm that finds the terminal vertex of an $l$-step-long path with probability $p$ and able to position non-blindly in every vertex of the path, requires either an exponential number (res) of restarts or an exponential cutoff as $l$ grows in length exponentially.

**Proof.** Suppose to have an oracle that, given a cutoff and an almost complementary solution, is able to say whether or not such a solution is farther than cutoff from the terminal vertex. If the solution is farther, then it is discarded, otherwise the algorithm follows the path from the given solution and the terminal vertex. The probability to find a randomly generated solution that is not farther than cutoff from the equilibrium by res random restarts is: $1 - (1 - \text{cutoff} / l)^{\text{res}}$. By posing: $1 - (1 - \text{cutoff} / l)^{\text{res}} = p$, we obtain $\text{res} = \log(1-p) / \log(1 - \text{cutoff} / l)$. When $\lim_{l \to +\infty} \text{cutoff} / l = 0$, we can write $\text{res} = - \log(1-p) / \text{cutoff}$. Therefore, if $l$ grows in length exponentially then either res or cutoff grow in length exponentially. When $\lim_{l \to +\infty} \text{cutoff} / l > 0$, cutoff grows in length exponentially as $l$ does.

From the above lemma, it can be easily observed that when all the LH paths grow in length exponentially non-blind randomization is useless: even dropping the completeness and accepting that the NE can be found with a probability $p$, in the worst case the compute time is $O(l^*)$. Thus, rrLH is asymptotically optimal among algorithms randomizing over LH paths.

### 3.2 Randomizing over Lemke (L) paths

As discussed in the previous section, randomization over paths exhibits a compute time that depends on the length of shortest path. The main drawback of LH is that the number of available paths is small. In this section, we resort to the Lemke algorithm adaptation as prescribed by [24], which we will call L. This algorithm allows for an arbitrary initial solution, each corresponding to a different path, and therefore it allows for an infinite number of paths. Define the polyhedron $P$ as follows.

$$P = \left\{ (z_0, z_1, z_2) \mid \begin{array}{c}
M_1, x_1 + M_1, x_2 + d_1, z_0 = q_1 \\
M_2, x_1 + M_2, x_2 + d_2, z_0 + q_2 = q_2
\end{array} \right\}$$

with $d_1 = 1, d_2 = \left[-(U_1, x_2)^T, -(U_1, x_1)^T\right]^T$ where $x_1$ and $x_2$ are parameters (therefore $P$ is parametric), $q_1 = -1, q_2 = 0$ and

$$z_1 = \begin{bmatrix} v_1 \\ v_2 \end{bmatrix}, M_{1,1} = \begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix}, M_{1,2} = \begin{bmatrix} 1^T & 0^T \\ 0^T & 1^T \end{bmatrix},$$

$$z_2 = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}, M_{2,1} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}, M_{2,2} = \begin{bmatrix} 0 & -U_1 \\ -U_2 & 0 \end{bmatrix}.$$

Let $V$ be the set of vertices of $P$. The space $\Theta$ of solutions traversed by the Lemke algorithm is a subset of $V$. Call $w$ the slack variables of $M_{2,1}z_1 + M_{2,2}z_2 + d_2z_0 + q_2 = 0$ and consider the associated tableau. Call $z_j$, with $j \neq 0$, the $j$-th element of $z = [z_1^T, z_2^T]$. Variables $z_j$ and $w_j$ are called complementary. A solution is completely complementary if the basis contains one complementary variable between $z_j$ and $w_j$ for every $j$ (therefore $z_0$ is out the basis), while a solution is almost complementary if both variables $z_j$ and $w_j$ for a single $j$ are not in the basis but $z_0$ is.

The algorithm moves along almost complementary solutions, each corresponding to a pair of strategies $(x_1 + z_0 x_1, x_2 + z_0 x_2)$. The initial solution is such that: (a) $z_0$ is in the basis and it is equal to 1, (b) all the variables $x_{i,a}$ such that $a$ is a best response to $x_{-i}$ are in the basis except one, and (c) for all the non-best-response actions $a$ the complementary variable $w_j$ of $x_{i,a}$ is in the basis. Given the initial solution, the algorithm follows a path of almost complementary solutions by repeatedly applying complementary pivoting (the entering variable is the complementary variable of the leaving variable at the previous step, while the leaving variable is determined by the lexico minimum ratio test). At the first step of L, the entering variable is $x_{i,a}$ such that $a$ is the only best response not yet in the basis. (L terminates when $z_0$ is the leaving variable, finding as NE.)

LH has a finite number of possible paths ($m_1 + m_2$), while L has an infinite number of them, thus we try to design a non-blind randomization policy among paths and we fixed a limit to the restarts, otherwise in the worst case the compute time would be infinite. We design the version of L with random restarts (rrL) like rrLH except that: (a) the initial solution is determined by randomly generating $x_1$ and $x_2$: since there are infinitely many possible initial solutions and many of them could lead to potentially long paths, (b) we use quality metrics to accept or discard an initial solution $\theta$ (we accept $\theta$ if $g(\theta) > th$, where $g$ can be defined in different ways, e.g., $\epsilon, \epsilon_{WS}$, and $r$, and $th$ is a threshold), (c) we use a cutoff as a function of how the different metrics $\epsilon, \epsilon_{WS}, r$, and $z_0$ decrease