A new parallel version of a dichotomy based algorithm for indexing powder diffraction data

Ivan Šimeček*, Aleksandr Zaloga† †Department of Computer Systems, Faculty of Information Technology, Czech Technical University in Prague, Prague, Czech Republic †Department of Chemistry, Siberian Federal University, Krasnoyarsk, Russia

(Original Paper)

Abstract—One of the key parts of the crystal structure solution process from powder diffraction data is the determination of the lattice parameters from experimental data shortly called indexing. The successive dichotomy method is the one of the most common ones for this process because it allows for an exhaustive search. In this paper, we discuss several improvements for this indexing method that significantly reduce the search space and decrease the solution time. We also propose a combination of this method with other indexing methods: grid search and TREOR. The effectiveness and time-consumption of such algorithm were tested on several datasets, including orthorhombic, monoclinic, and triclinic examples. Finally, we discuss the impacts of the proposed improvements.

Index Terms—indexing powder diffraction, dichotomy algorithm, TREOR algorithm, grid search algorithm.

I. INTRODUCTION

The unit-cell dimensions (also called lattice parameters) \( (a, b, c, \alpha, \beta, \gamma) \) determine the positions of peaks in a powder diffraction pattern. The aim of indexing is to determine the correct lattice parameters by using the information on the peak positions in the experimental pattern. Although there are many implementations, the number of existing methods is relatively small:

- Monte Carlo method (simulated annealing search) — McMaille [1].
- ITO [2].
- SVD-Index [3].
- genetic algorithm [4].
- Conograph [5], [6].
- trial-and-error methods — TREOR [7], [8].
- grid search [9].

This paper aims to improve mainly at the last one method (and the combination of the last three methods). This method represented by two main implementations has been continuously improved from its beginning to the current state, for example, DICVOL package has evolved into DICVOL91 [12], then into DICVOL04 [13], then into DICVOL06 [14], then into DICVOL14 [15], but even the newest version inherit some drawbacks of their predecessors.

A. Related works

As far as we know, the most related software to our work, X-Cell and DICVOL14 traverse the search space inefficiently, so it is difficult to use them in very complex indexing situations (high number of spurious lines or low symmetries).

B. Used technologies

Brute force algorithms need significant computing power. Such power can be attained by parallel computing and the paper considers shared memory environment with OpenMP [16], [17]. OpenMP is a cross-platform standard for parallel processing. The OpenMP API specification is defined as a collection of compiler directives, library routines, and environment variables extending the C, C++, and Fortran programming languages. They can be used to create portable parallel (multi-threaded) programs utilizing shared memory.

II. CURRENT STATE-OF-ART

A. Description of the powder pattern

The measured powder diffraction pattern represents the reciprocal space and contains information about dimensions of the reciprocal unit cell. Each position of the observed diffracted line \( \theta_{hkl} \) corresponds to one or more inter-planar distance \( d_{hkl} \) of the reciprocal space, where \( h, k \) and \( l \) are Miller indices (small integer numbers). The relationship between the inter-planar spacing \( d_{hkl} \) of the crystal lattice and the position of the diffracted line \( \theta_{hkl} \) in the powder pattern is ruled by Bragg’s law:

\[
 n\lambda = 2d_{hkl} \sin \theta_{hkl},
\]

where
- \( n \) is an integer that denotes the order of reflection (it is usually assumed to be 1),
- \( \lambda \) is the wavelength of the incident ray,
- \( d_{hkl} \) is the inter-planar spacing.

Let \( a^*, b^*, c^*, \alpha^*, \beta^*, \gamma^* \) denote the parameters of the reciprocal unit cell. The relationship between these parameters and the inter-planar distance \( d_{hkl} \) is:

\[
 1/d_{hkl}^2 = a_{11}h^2 + a_{22}k^2 + a_{33}l^2 + 2a_{12}hk + 2a_{23}kl + 2a_{13}hl,
\]

where \( a_{11} = (a^*)^2, \quad a_{22} = (b^*)^2, \quad a_{33} = (c^*)^2, \quad a_{12} = 2a^*b^* \cos \gamma^*, \quad a_{23} = 2b^*c^* \cos \alpha^*, \quad a_{13} = 2a^*c^* \cos \beta^* \).
B. Terminology and notation

The parameter $N$ denotes the number of input reflexes. The parameter $DOF$ denotes the number of unknown lattice parameters (degree of freedom). Depending on the crystal system, some values $a_{ij}$ are zero. For example, in the cubic system, the expressions (1) and (2) are simplified to:

$$Q_{hkl} = (h^2 + k^2 + l^2)/a^2.$$  

The parameter $\text{max}_imp$ denotes the number of impurities in the input reflexes.

To distinguish between different types of values of $Q$, we further denote by $Q^{calc}$ only the values of $Q$ obtained from the theoretical model and by $Q^{obs}$ the measured (observed) values of $Q$.

A box $B \subseteq \mathbb{R}^s$ is defined as a product of closed real intervals. Thus, every box is characterized by $2^s$ real numbers $B_1^{L,\ldots,U}$. $B_k^{L}$ is the lower bound in $k$th dimension, and $B_k^{U}$ is the upper bound in $k$th dimension. We assume only nonempty boxes, thus $\forall k \in \{1 \ldots s\}, B_k^L < B_k^U$.

III. METHODS FOR INDEXING

A. Grid search method

The grid search method is the simplest brute force indexing method based on systematic testing of search space. The following study [18] had demonstrated that brute force algorithms could better solve some complex problems in comparison to the old fine tuned sophisticated algorithms.

B. TREOR method

The basic idea of the TREOR method (for details see [7], [8]) follows: the reciprocal cell relationship from Eq.[2] may be written as

$$M \bar{A} = \bar{Y},$$

where

- $M$ is a square matrix containing the Miller indices (derived from $hkl$ triplets),
- $\bar{A}$ is a vector containing the parameters of the reciprocal unit cell,
- $\bar{Y}$ is a vector with $DOF$ values from $Q^{obs}$.

The dimensions of all vectors and matrices are equal to $DOF$, and the process of indexation can be solved by this SLE, but the matrix $M$ and the vector $\bar{Y}$ are unknown, so we must try all possible permutations.

C. Successive dichotomy method

The successive dichotomy (SD) method can be viewed as an application of interval arithmetic [19], [20] on the indexing process.

The key idea of the SD method is simple:
1) It starts with one initial box that is derived from the user-defined intervals of unit cell parameters.
2) The current box is tested if it can or cannot contain solutions.
   - If it cannot, then the current box (called “unfavourable volume element”) is rejected from further investigation.
   - If it can, then the current box (called “favourable volume element”) is divided into smaller boxes and assigned for further investigation.
3) The whole process (point 2) recursively repeats until one or all solutions are found.

The whole process of the SD method can be illustrated by a dichotomy tree (DT). To precise the aforementioned algorithm, we define the following functions:

- $\text{Generate}(B, \text{HKL set})$ that for the given box $B$ and the set of possible $hkl$ trials ($\text{HKL set}$) returns the set of $Q$.
- $\text{Verify}(Q^{calc}, Q^{obs})$ that for the given sets $Q^{calc}$ and $Q^{obs}$ verifies whether or not the $Q^{calc}$ may contain an indexing solution i.e., at least $n - \text{max}_imp$ intervals of $Q^{obs}$ is overlapped by intervals from $Q^{calc}$. This function returns a logical value.
- $\text{Divide}(B)$ that for the given box $B$ returns the set of sub-boxes (dissection of the box $B$).
- $\text{SmallEnough}(B)$ that for the given box $B$ returns if the widths of the box $B$ are “small enough”. This function returns a logical value.
- $\text{CheckForSolution}(B)$ that for the given (small) box $B$ returns if the box is the indexing solution. This function returns a logical value.

With the mentioned functions, the algorithm for SD can be rewritten as follows:

Algorithm 1: Dichotomy base idea

1: procedure DICHOTOMY1
2: Input: $B =$ initial box
3: Input: $\text{HKL set} =$ set of possible $hkl$ trials
4: Output: $solution =$ set of the indexing solutions
5: 2: push $B$ onto the stack
6: 3: while $solution$ is not found AND stack is not empty do
7: 4: pop $B$ from the stack \( \triangleright B \) is the current box
8: 5: $Q^{calc} \leftarrow \text{Generate}(B, \text{HKL set})$
9: 6: if $(\text{Verify}(Q^{calc}, Q^{obs})=\text{true})$ then \( \triangleright B \) can contain solution
10: 7: \hspace{1cm} if $(\text{CheckForSolution}(B)=\text{true})$ then
11: 8: \hspace{1cm} \hspace{1cm} $solution \leftarrow solution \cup B$
12: 9: \hspace{1cm} else
13: 10: \hspace{1cm} \hspace{1cm} $B_{\text{new}} \leftarrow \text{Divide}(B)$
14: 11: \hspace{1cm} \hspace{1cm} push all $B_{\text{new}}$ onto the stack
15: 12: \hspace{1cm} return solution or “solution is not found”
IV. Our New Improvements

In this section, we discussed the impact of our improvement (VI) to existing implementations (X-Cell or DICVOL) proposed in Section III-C2.

A. Experiments configuration

We have implemented all algorithms in C/C++ using OpenMP.

1) HW and SW configurations:

- Testing configuration 1: Some experiments were performed on a small university cluster called “star”. The execution times were measured on a server with following HW and SW parameters:
  - 2 × CPU Intel Xeon Processor E5-2620 v2,
  - CPU cores: 6 per CPU, 12 in total,
  - Memory size: 32 GB RAM,
  - OS Linux, C++ compiler (g++) version 4.8.3

2) Indexing data files: The following data files were used:

- Orthol: test3b.dat (B₃OH₂NO₃, CIF 4065373), N = 20, the orthorhombic crystal system, the correct solution: a = 11.304Å, b = 11.02Å, c = 9.152Å, published in [21].
- Ortho2: testlib.dat (Cd₃(OH)₅(NO₃)), N = 20, the orthorhombic crystal system, the correct solution: a = 3.4203(3)Å, b = 10.0292(6)Å, c = 11.0295(6)Å, published in [22].
- Mono1: cim.dat (Cimetidine, C₁₂H₁₆N₆S), N = 21, the monoclinic crystal system, the correct solution: a = 6.821(1)Å, b = 18.818(3)Å, c = 10.374(2)Å, β = 106.42(1)°, published in [23].
- Mono2: CSA.dat (TGRIC : Cyclosporin A dimethylsulfoxide solvate, Deposition Number: 1268484, N = 24, the monoclinic crystal system, the correct solution: a = 15.521(2)Å, b = 20.833(3)Å, c = 12.949(3)Å, β = 100.21(1)°, published in [24].
- Mono3: Taxol.dat (C₄₅H₉₅NO₁₃ · 3(C₄H₆O₂), N = 20, the monoclinic crystal system, the correct solution: a = 16.329(2)Å, b = 17.704(2)Å, c = 17.504(1)Å, β = 100.61(1)°, published in [25].
- Tril: cap_3-100_mcmv.dat (C₂₂H₂₈CuIN₂O₄, N = 48, the triclinic crystal system, the correct solution: a = 5.164309Å, b = 9.46518Å, c = 11.38433Å, α = 73.17°, β = 89.1931°, γ = 87.91°).

3) Initial parameters: We have used the following values as initial:

- all lengths are between 3Å and 30Å.
- all angles are between 90° and 120°.
- minimal cell volume $V_{\text{min}} = 30 \text{Å}^3$ and maximal cell volume $V_{\text{max}} = 3000 \text{Å}^3$.
- initial hkl-triplet set contains 1000 elements.

B. Evaluation of general results

In this Section, we evaluate general results of the successive dichotomy method.
1) Impact of the number of lines: We can change the number of input lines. The results are shown in Figure 2. Obviously, with growing \( N \) (i.e., more conditions are checked within \( \text{Verify}(Q_{\text{calc}}, Q_{\text{obs}}) \) in procedure \( \text{Dychotomy1} \), we get fewer nodes in the DT.

2) Impact of the number of HKL trials: We can change the number of assumed initial \( hkl \) trials. For orthorhombic and monoclinic cells, the values (the maximal values of \( hkl \)-triplets) used for time measurement are much higher than the minimal possible values. The results are shown in Figure 3. Obviously, with growing \(|hkl|\) (i.e., it is easier to satisfy conditions within \( \text{Verify}(Q_{\text{calc}}, Q_{\text{obs}}) \), we get more nodes in the DT.

3) Impact of the “first lines” optimization: We have also measured the effect “first lines” optimization, but as was discussed in Section III-C1, we don’t include this optimization. If we apply the following restrictions on the first 9 lines (reflexes) on \( \text{mono1} \):

\[
h, k \leq 3, \quad |l| \leq 1, \quad h + |k| + |l| \leq 4
\]

Results are shown at Figure 4. With “first lines” optimization we get 20 times less nodes at depth= 30.

If we apply the following restrictions on the first 6 lines (reflexes) on \( \text{tril} \):

\[
h \leq 1, \quad |k| \leq 1, \quad |l| \leq 2, \quad h + |k| + |l| \leq 2
\]

Results are shown at Figure 5. With “first lines” optimization we get 5 times less nodes at depth= 30.

C. Search space

As far as we know, DICVOL uses for searching the direct space, X-Cell uses the reciprocal space. The big advantage of real space usage is the transparency to the user and the fairness of dissection. Let us assume an example of a cubic system for an explanation of this phenomenon. We know that the correct solution \((a)\) is in \([2, 30]\). We also assume that the probability that any value in this interval is equal to the correct solution is the same along with the whole interval.
If we divide the starting interval in real space, we get two intervals \([2, 16]\) and \([16, 30]\) with the same probability that the solution lies in. If we divide the starting interval in reciprocal space \([2, 30]\) is transformed to \([1/30^2, 1/2^2]\), we get two intervals \([1/30^2, 113/900]\) and \([113/900, 1/2^2]\) that is equal to \([2.2, 8.22]\) and \([2.822, 30]\) in real space. So, the probability for the first interval (that the solution lies in) is about 2.9% and about 97.1% for the second interval.

On the other hand, the advantages of reciprocal space are the following:

- It eliminates the overhead of transformation (from real to reciprocal and vice versa) because reciprocal values are needed to decide if the box would be rejected or not in function \(VERIFY(Q^{\text{calc}}, Q^{\text{obs}})\).
- It “converges” more quickly because transformation (from real to reciprocal) increases the “volume” of boxes. For the demonstration, let us assume an example of a monoclinic system, the input box \(B\) with the parameters \(a, b,\) and \(c\) are in \([2, 10]\), and \(\beta\) is in \([45^\circ, 90^\circ]\). The dissection in real space results in 16 boxes. We will discuss further only the first two boxes: \([2, 6] \times [2, 6] \times [2, 6] \times [45^\circ, 67.5^\circ]\) and \([2, 6] \times [2, 6] \times [2, 6] \times [67.5^\circ, 90^\circ]\). Since \(a_{11} = \frac{\sqrt{2} \sin \beta}{4}\), these two intervals are transformed into two intervals in reciprocal space for \(a_{11} = 0.03254, 0.5\)
and \([0.02778, 0.2928]\). Please note that these intervals are not disjoint. Therefore, the efficiency of searching is decreased.

Based on this observation, we decide to mix both approaches: on the top level, we dissect (divide) boxes in real space (to satisfy fairness of dissection), and in deeper levels, we divide boxes in reciprocal space (to increase the efficiency of traversing).

Table I illustrated the impact of this mixed strategy on the number of nodes of DT. The optimal value is if the number of layers (dissected in real space) is approximately equal to the number of layers (dissected in reciprocal space).

### 1) Sensitivity of indexing
Since it is very heavy or impossible to derive the value of the proper maximal depth of DT, we have done an experiment:

- 3 data files, each with different crystal system (Orthol, Monol, Tril).
- We know the exact correct solution for each data file.
- We want to evaluate the size of “favourable volume element” in every dimension: We start with correct parameters. Then, we decrease or increase cell parameters until the current box is changed into “unfavourable volume element". In other words, \(DL\) and \(DU\) represent the maximal difference from correct solution while unit cell parameters are still valid, i.e., all lines are indexed.

Tables I and II can be used to estimate the proper maximal depth of DT (denoted by \(depth_{\text{min}}\)) with dissection by the following equation:

\[
depth_{\text{min}} = \log_{\text{arity}} \prod_{i} U(p_i) - L(p_i)/DL(p_i) + DU(p_i), \quad p_i \in \{a_{11}, \ldots, a_{33}\}
\]

1Strictly speaking the intervals in reciprocal space are not also disjoint but they share only one point.

where

- \(arity\) is the arity of the DT.
- The results for real space and \(arity = 2\):
  - \(depth_{\text{min}}\) (orthorhombic) = 22.7,
  - \(depth_{\text{min}}\) (monoclinic) = 27.1,
  - \(depth_{\text{min}}\) (triclinic) = 38.2.

The results for reciprocal space and \(arity = 2\):

- \(depth_{\text{min}}\) (orthorhombic) = 29.1,
- \(depth_{\text{min}}\) (monoclinic) = 36.4,
- \(depth_{\text{min}}\) (triclinic) = 52.8.

These values are rather optimistic since we change each parameter separately. But we can conclude that the SD method is (without further optimizations) unreal for a triclinic case. And we can also conclude that the depth of DT in real space is lower than in reciprocal. This is another significant reason to use a mixed strategy (see Section IV-C).

### D. Reduction of intervals
In our implementation of function \(VERIFY(Q^{\text{calc}}, Q^{\text{obs}})\), we not only test the box would be rejected or not, but we also try to reduce the widths of intervals. It can be done if one reflection \(Q_{\text{obs}}\) is overlapped by exactly one \(hkl\) trial (we call this situation a unique overlap). For the sake of simplicity, we assume that the crystal system is orthorhombic, so the following statement is valid:

\[
[a_{11}, a_{11}^H] h^2 + [a_{22}, a_{22}^H] k^2 + [a_{33}, a_{33}^H] l^2 \in [q_{11}, q_{11}^H]
\]

Under the assumption that \(h \neq 0\), we can compute temporary values of \(a_{11}^L\) and \(a_{11}^H\) as follows:

\[
temp_{a_{11}^L} = a_{11}^L - a_{22}^L k^2 - a_{33}^L l^2 / h^2
\]

\[
temp_{a_{11}^H} = a_{11}^H - a_{22}^H k^2 - a_{33}^H l^2 / h^2
\]

and similarly for \(temp_{a_{22}^L}, temp_{a_{22}^H}, temp_{a_{33}^L},\) and \(temp_{a_{33}^H}\). Then, we can try to reduce intervals:

\[
[new_{a_{11}^L}, new_{a_{11}^H}] = [temp_{a_{11}^L}, temp_{a_{11}^H}] \cap [a_{11}^L, a_{11}^H]
\]

The second effect of unique overlap is caused by the fact that every \(hkl\) trial can index at most one reflection. Consequently, if for \(q_{i}\) unique overlap by the \(hkl\) trial occurs, then this \(hkl\) trial is removed from the \(HKL\) set for all other lines.

### E. Early solution
The existing implementations try to solve the box \(B\) only if the maximal level of dichotomy tree is reached or if \(B\) is small enough (function \texttt{SmallEnough}(B) in Algorithm I). We additionally try to solve if the number of unique overlaps is equal or higher to \(DOF\), so these unique overlap construct system of linear equations (SLE) in the form (for an orthorhombic system):

\[
h^2(a_{11}^L + a_{11}^H)/2 + k^2(a_{22}^L + a_{22}^H)/2 + l^2(a_{33}^L + a_{33}^H)/2 = Q_{\text{obs}}
\]

We use Gramm-Schmidt or LSQ method to solve such SLE. It corresponds to TReOR indexing algorithm (see Sec. III-B). If this method successfully finds the solution (i.e., the number
of linearly independent equations is equal to DOF) then we test if the solution is the correct indexing solution and cancel this branch of DT.

F. Different approach to dissection

We have also observed that for some boxes, the dissection does not reduce the numbers of hkl trials overlapping q intervals and methods mentioned in Sections IV-D or IV-E cannot be applied. These boxes usually reached the maximal depth of recursion (and the number of nodes in the dichotomy tree grows rapidly). The alternative solution (in this situation) is that we do not dissect the box, but we construct the child nodes as unique overlaps. For example, we assume that one line $q_i$ is overlapped by 3 hkl trials: $(h_1, k_1, l_1)$, $(h_2, k_2, l_2)$, and $(h_3, k_3, l_3)$. The next level of dichotomy tree consists of 3 child nodes: in the first node, the line $q_i$ is assumed be overlapped by only $(h_1, k_1, l_1)$ trial, and so on. Since the unique overlap has two effects (see Section IV-D), it can reduce the number of nodes that reached the maximal depth of recursion.

G. Results of our improvements

Table IV illustrated the number of nodes of the dichotomy tree without and with our improvements (VIs), the sum of the number of nodes with VIs is approximately 10 times less than without them.

H. Using volume intervals

The search-space can be dramatically reduced also if the interval of unit cell volume is known. There are some methods for estimation of unit cell volume (see [20], [27], [28]), but these methods can fail, and we lost the main property of the dichotomy method: exhaustive search. Therefore, we use the same method that is commonly used in the crystallography.
community: we divided the initial unit cell volume interval (VI) into smaller disjoint intervals (sometimes called “volume slices”). Then, we check them in ascending order until the solution is found. Table VI illustrated the number of nodes (of dichotomy tree) without and with VIs, the sum of the number of nodes with VIs is approximately 10 times less than without them.

I. Comparison with DICVOL and X-Cell

Our program for indexing is called ParaCell and it can be obtained from this URL [29]. It has tolerance for impurity lines, and may search for a zero-point (see details in the readme file on the program website). It support multiple method of indexing, for example TREOR method [8].

Straightforward comparison with other implementations of SD method is not possible: e.g., number of nodes is an internal information, we are unable to get this information from DICVOL or X-Cell program, some important program settings (e.g. used hkl-triples) of DICVOL and X-Cell are not known, program usually try to find any solution with the smallest volume, etc.

The main difference to the previous versions of the SD algorithm is in the basic idea that the program simply searches all combinations instead of trying to apply the crystallographic algorithm is in the basic idea that the program simply searches smallest volume, etc.

J. Comparison with other indexing software

For comparison with all well-known indexing software we use Powder Diffraction Indexing Benchmarks [30].

Benchmarks consist of 10 indexing tests, result of each test can be number +1,0, or −1:

- The 1 point note to tests means that the correct cell was found in first FoM position among the proposals.
- The zero point note means that the correct cell is mixed with uncorrect ones, not at the head of the list. The order of the true solution in the list is given as a subscript : 06 means that it was the sixth cell proposal. But the order has to be < 10, otherwise, there is a -1 point note given.
- The -1 point note means that the correct cell was not found at all, or at a position larger than 10 in the lists.

The results of our program are in Table VI results of other indexing software are in Table VII.

Our setup: Manual mode, dichotomy method is chosen, max. 5 unindexed lines are tolerated. The peak position error tolerance is 0.15°, exploration is made only in monoclinic in a restricted cell volume range (800 – 1200Å³). That version has tolerance for impurity lines, and may search for a zeropoint.

Overall results of our program (with dichotomy method) are quite satisfactory.

V. Conclusions

We found the idea behind the successive dichotomy rewarding. Our program is not an extension to the existing programs, it was build up from the ground. In the current version, it is a “pure” implementation of the successive dichotomy idea. There has remained an interest in extending this idea further to better handle data in low-symmetries (the monoclinic and triclinic) systems. In this paper, we propose some optimizations to reduce the indexing search space by using three indexing methods.

ACKNOWLEDGMENTS

This work was supported by the Grant Agency of the Czech Technical University in Prague, grant No. SGS20/212/OKH3/3T/18.

REFERENCES

TABLE VI

<table>
<thead>
<tr>
<th>A1</th>
<th>A2</th>
<th>B1</th>
<th>B2</th>
<th>C1</th>
<th>C2</th>
<th>D1</th>
<th>D2</th>
<th>E3</th>
<th>F4</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>+1</td>
<td>+1</td>
<td>+1</td>
<td>+1</td>
<td>+1</td>
<td>0_b</td>
<td>+1</td>
<td>+1</td>
<td>+1</td>
</tr>
</tbody>
</table>

TABLE VII

<table>
<thead>
<tr>
<th>Program</th>
<th>score</th>
<th>note</th>
</tr>
</thead>
<tbody>
<tr>
<td>TIO7</td>
<td>-6</td>
<td></td>
</tr>
<tr>
<td>DICVOL91</td>
<td>-2</td>
<td></td>
</tr>
<tr>
<td>DICVOL04 (a)</td>
<td>+1</td>
<td>fast default and manual tests made by A. Le Bail</td>
</tr>
<tr>
<td>TREOR90</td>
<td>-2</td>
<td></td>
</tr>
<tr>
<td>DICVOL04 (b)</td>
<td>+9</td>
<td>manual tests made by D. Louër</td>
</tr>
<tr>
<td>McMaille</td>
<td>+7</td>
<td></td>
</tr>
<tr>
<td>ParaCell</td>
<td>+5</td>
<td></td>
</tr>
</tbody>
</table>


