Decision making by heterogeneous cell populations: immune-tumor interactions under metronomic chemotherapy and distributed computation in synthetic biology

A Thesis Presented

By

Anh Phong Tran

to

The Department of Chemical Engineering

in partial fulfillment of the requirements for the degree of

Doctor of Philosophy

in the field of

Chemical Engineering

Northeastern University
Boston, Massachusetts

December, 2020

Acknowledgments

I owe much of this work, my academic contributions, and my personal development to Professor Eduardo D. Sontag. As much as I want to believe that one has much control over how to shape one's life, I was fortunate that our paths crossed and I was blessed with his constant support throughout the last two years together. There is no doubt I owe much of my growth as a scientist to Eduardo's insatiable quest for knowledge and scientific discoveries as well as his very distinctive attention to details that will continue to inspire my academic journey. Yet, there is nothing that has changed my life more than his decision to extend a hand when I needed it most and, for that and countless other things, I will always be grateful.

Many thanks go to my committee members, Professors Edgar D. Goluch, Steve Lustig, and Dana Brooks for their continued support throughout the dissertation process. In particular, I wish to emphasize the importance of Professor Goluch on ensuring that I could fulfill the degree requirements, the helpful discussions with Professor Lustig as well as Professor Brooks's feedback and availability despite the challenging circumstances of 2020.

Much of my interest in pursuing research in applied mathematics started with Professor Jerry H. Meldon, something that I am incredibly grateful for. The influence of Professor Christos Georgakis with whom my first paper was published must also be acknowledged with all the important lessons that come with it. Professor Steve Jacques's vision about the field of light optics will continue to be an inspiration and I am hopeful that our work together will one day be of importance. Also, while our views differed in various ways, my time in Professor Qianqian Fang's lab undeniably

contributed to numerous of my research interests.

I would also like to thank my fellow lab members and collaborators. Notably, Dr. M. Ali Al-Radhawi has been a particularly strong influence in helping me accomplish a lot of the work in this thesis. I could always count on his mentorship and brilliance to navigate difficult problems. We also often discussed more than just research and I enjoyed those moments very much as well. Shu Wang and Mahdiar Sadeghi have also been supportive during my time in the lab and they helped me navigate some of the more difficult times in the life of Ph.D. student. I am also fortunate to have met, now Professors, Cynthia Sanchez Tapia, and James Greene with their constant willingness to share their academic experiences. I am grateful for having collaborated with Dr. Irina Kareva, who suggested and initiated the project about modeling metronomic chemotherapy. I also received important help from members of Professor David Waxman's group including himself, Dr. Junjie Wu, and Professor Joshua Doloff. I would also like to thank other collaborators such as Professor Christopher Voigt, Professor Elisabeth Ernst, Dr. Paolo Cassano, and Shijie Yan.

Much love and thanks to my mother, Ngo Thi Tuong Van, who sacrificed everything to raise me and my brother. She always gave us all that she had so that we could have the chances she did not. Many thanks must also go to my aunt, Thai An Smolski, and my uncle, John Smolski, who supported me throughout my studies. My aunt is my constant reminder of what it means to display mental fortitude as she has been navigating her cancer these last few years. To my mother and my aunt, I dedicate this thesis. There are other family members that I must acknowledge for their various contributions: my aunts Thai Anh Ngo and Nyno Broucke, my aunt's husband, Michel Broucke, my uncle, Vinh Hai Ngo, his wife, Thuy Hoang Dac Ho, my uncle, Ngo Ving Long, as well as my maternal grandfather for emphasizing the importance of education. I would like to also thank my brother, Anh Phi Tran, for sticking with me through the years and my numerous cousins and other family members that I have not mentioned by name here.

Lastly, but not least importantly, I could not have made it through graduate school without the friends that I have made along the way. In particular, I must

thank Robert Dimatteo for being always there for me through the highs, the lows, and for the many adventures that we have had together. My friendship with Namson Ngo-Le has never ceased to enlighten my views of the World. Minh Trinh, in addition to being a great friend, has always been there for me when I faced my most difficult decisions. I would like to thank the many friends from Vietnam: Nga, Nhu, Duy, Quyen, Tam, Dzung, Thao, Trang, Long, Minh, Phi, Hoang, Luna, Tri, Hien Anh, Ha Anh, among others. There are also many important friendships from my time at Northeastern: Stanley, Hunter, Wenjin, Dat, Andres, Artem, Xin, Rashida, and Pravin. Finally, I would like to acknowledge people from other walks of life that have had a meaningful impact: Denzel, Abdurrahman, Leo, Long, Jeff, Marco, Kailin, Pierre, Damien, Nicolas, Paul, Jed and Agnes, those who kept me entertained during the COVID-19 pandemic, the family of Professor Meldon, and those that I have forgotten to mention here but played an important role in my life.

List of Tables

List of Figures

Contents

Abstract

This work concerns itself with the development of improved therapies in cancer through a better quantitative understanding of the biological mechanisms that can enhance the anti-tumor immune response during metronomic chemotherapy and the development of improved synthesis algorithms to implement greater logical complexity in engineered cells through the idea of distributed computation.

Under the right drug dose and treatment schedule, metronomic chemotherapy, where a smaller dose than the maximum tolerated dose is given intermittently, has shown the potential to initiate and sustain a prolonged anti-tumor immune response. The biological mechanisms remain incompletely understood and an important emphasis of this work is to use experimental data, in which mice were implanted glioma and glioblastoma tumors, to develop a mathematical model that fits the observed dynamics of the tumor volume. The model includes terms representing immune recruitment as well as the emergence of drug resistance during prolonged metronomic treatments. Notably, this work analyzes the role of drug resistance, unravels the contributions of various immune components, and describes the complex interactions of the immune system with the tumor and the drug.

There have been important recent developments in the domain of immunotherapy using engineered cells, with outstanding results of total remission in some patients. Yet, the lack of specificity or control over the behavior of these cells has also led to some catastrophic results that have reinforced the need for engineered cells to make more informed decisions. The second part of this work focuses on the development of efficient logical circuits under the constraint provided by the number of implementable "gates" per cell. This limits implementing more gates in a cell. To alleviate this

limitation, a synthesis methodology is developed to exploit the ability of cells to communicate using diffusible molecules such as quorum sensing chemicals among bacterial colonies or cytokine signaling among immune cells. Thus, by partitioning pieces of an overall logical circuit among multiple cells, one can implement larger circuits under biological constraints through the idea of distributed computation.

Chapter 1

Overview and Motivation

With an aging population, cancer is now the second leading cause of death in the US and in the world [8, 9]. In recent years, there have been several significant advances in the fight against cancer. The democratization of genome sequencing has allowed a better understanding of the role that genes play on certain cancers [10]. It also allowed to find populations at risk of certain types of cancer, as well as help finding more personalized treatments. A promising class of treatment called immune checkpoint inhibitors that target T cells has been developed to enhance the immune response of the host against cancer cells [11]. Stem cell transplants are being increasingly used to restore blood-forming stem cells in patients that have been treated with high doses of chemotherapy or radiation therapy [12, 13]. These new treatments show how our understanding of how cancer operates and how it is to be treated is continuously evolving. Addressing the whole complexity and various ways in which cancers can manifest themselves is an insurmountable task, and the primary aims of this work were focused on two main ways of enhancing the immune response to cancer.

The first aim of this work is to provide a better understanding of the mechanisms driving metronomic chemotherapy. This work was developed based on extensive experimental data on mice using cyclophosphamide given under this metronomic regimen [14, 3, 15, 16, 17]. It has been understood for a number of years now that injecting lower doses of chemotherapy drugs with enough breaks in between treatments can, in some cases, lead to an increase in anti-tumor immune activity [18].

However, a lot about the mechanisms driving metronomic chemotherapy remains a mystery. The early promising results in animal models did not quite translate to breakthroughs during clinical trials, and this approach has not yet supplanted more traditional chemotherapy regimens such as maximum tolerated dose (MTD) and daily dose regimens. What makes this process particularly difficult to elucidate is the importance of treating the host in a narrow window that balances the drug dose and the timing of the injections that allow for an anti-tumor immune response to be elicited [19, 20]. Not only is it already exceedingly difficult to understand the complex biological interactions between the treatment, the tumor, and the immune system, but one also needs to drive the right momentum to be able to see that desired effect.

Part I centers around how to create a dynamic model that is an interpretation of the action that an induced cell death drug such as cyclophosphamide has on a tumor when given on a metronomic regimen. An important and consistent observation in successful metronomic chemotherapy treatments in mice showed that the immune cells initially see a significant decrease in their numbers early on after an injection of the drug. However, what ensues is that, if the doses are moderate and there is enough of a break between injections, the immune response can be observed to not only come back over time but oftentimes, exceed the untreated response by an order of magnitude [3, 15]. These dynamics were modeled by employing a common biological network motif called an incoherent feedforward loop (IFFL) that allows for a decrease early on in the immune response, but an overall increase in the longer-term [21]. While there have been attempts to develop models of metronomic chemotherapy [22, 23, 19], they fail to address how to properly model the observed dynamics surrounding the immune recruitment. The significance of the work presented in this chapter resides in the fact that a wide range of experimental conditions is fitted accurately using only one set of parameters while also proposing a very compact model of only five differential equations. In addition to modeling the dynamics of the immune system, the model also contains a component that deals with the increase in drug resistance over time that results from the repeated treatments [24]. Finding an appropriate mathematical description of how injecting a drug like cyclophosphamide can lead to

both a drop and an increase in immune cells under certain conditions is important not only because it fits the data well but because it emphasizes the importance of the dynamical nature of biological interactions.

Part I also emphasizes developing a better description of the understanding of how an anti-tumor immune response is being elicited and what conditions are required for it to occur. Besides the structural implications that the IFFL-based model has on better understanding the interactions between the tumor, drug, and immune system, the added value of an accurate mechanistic model lies in its ability to potentially inform better treatment regimens that may seem, at first, counter-intuitive. However, for the model to be truly predictive, there was a need to create a more generalized model that also accounts for regimens during which the immune system isn't being recruited. The first model performed particularly well when the breaks between drug injections were spaced out by 6 to 12 days [3]. Yet, when looking at experimental data in which drug breaks were spaced every day or every three days, there is no observed immune recruitment. Thus, for the proposed model to be truly predictive, it also needs to describe well what happens when the treatments are given "too frequently" to be able to elicit an anti-tumor immune response. The experimental data on SCID mice, devoid of B and T cells, unequivocally showed that treatments given every three or fewer days led to insufficient recruitment of immune cells, and anti-tumor factors. This persistent problem is believed to be linked to the IFFL idea, or rather, the lack thereof.

The IFFL motif of primary interest is when the introduction of the drug leads to the recruitment of an immunostimulatory intermediate that will over time lead to the recruitment of the immune response. A likely candidate for such intermediate is hypothesized to be type I interferon as shown in the work of Du and Waxman [25]. Another potential scenario would be that the injection of cyclophosphamide causes immunosuppressive factors "like" regulatory T cells (but for the innate immune system) to decrease in numbers, allowing for an increase in immune cell recruitment [26]. Let's also note that in SCID mice, the anti-tumor immune response is vital to tumor reduction, as the cytotoxicity of the drug alone is insufficient to control

the growth of the tumor. Experimental data indicates that the first event after an injection is a reduction in immune cells present in the tumor microenvironment due to the cytotoxicity of the drug [3]. It is followed, in the case of type I interferon, by an increase of this immunostimulatory factor until around 3 days after the injection where it peaks [25]. It is believed that allowing for breaks of 6 days or more in between injections leaves enough time for an intermediate such as type I interferon to fulfill its role in recruiting an anti-tumor response. Thus, the first step was to find out an appropriate functional form that would describe this interference on the immune recruitment of successive injections in close temporal proximity.

Using the gene expression data of various immune markers that include those of NK cells, dendritic cells, and macrophages, it was found that the immune recruitment is very large in magnitude compared to the normal activity of these cells but also that, when recruited, the activity of these cells tends to increase in unison, leading to a strong tumor regression. One of the primary challenges was due to the necessity of connecting datasets that not only involve different treatments conditions, but also data that are based on various tumor cell lines, different initial volumes of the tumor at the start of the treatment, or the use of mice with different immune profiles, in addition to the usual variability between individual experiments. Yet, it was found that thinking in terms of how the phenomenological representations correlate with the data was more important than matching the raw magnitudes of these changes. In practice, matching all the different cell types, cytokines, and chemokines would simply be too daunting of a task in terms of required parameters and equations that would need to be fitted. The gene expression data for the immune cells also happens to be too sparse to allow such fitting to be done properly. Using the correlation approach, the model structure was improved to represent the different observed behaviors for immune recruitment.

The primary focus of this first part was on how to get an ICD drug like cyclophosphamide to maintain a strong anti-tumor immune response over a prolonged period of time to keep the tumor at bay. This step of designing more effective treatments is an important one because many of these chemotherapy drugs have already been

approved for use by the FDA and have been a standard of care for cancer for decades [27]. Thus, there is a strong interest in understanding how regimens such as metronomic chemotherapy and some of the complex treatment schedules that are proposed here can improve the efficacy of existing therapies. Yet, while chemotherapy is used as a targeted treatment towards killing cancerous cells, or more generally fast-dividing cells, it is not remarkably specific to cancer cells and comes with side effects and complications that can be severe in nature. The vast array of targets that chemotherapy drugs affect is an important limitation to the extent that one can leverage the cytotoxicity of the drug and the host immune system to fight off cancer. However, cancer cells often have tumor antigens or specific markers that differentiate them from normal cells. There have been a number of novel immunotherapy treatments that try to stimulate specific components of the immune system and in many cases can or are used in conjunction with chemotherapy drugs. A relevant example of this is in the context of metronomic chemotherapy of cyclophosphamide used with a CpG-1826 immunotherapy treatment, where it was shown that the combined treatment can potentiate chemotherapeutic and anti-tumor immune responses [28].

The second part of this work focuses on the distributed computation among immune cell populations. The immune system is composed of various cell populations that constantly communicate to exchange information and make complex decisions about possible threats. These cells individually perform logic operations as part of a larger distributed network of biological circuits. Engineering the immune system to alter its decision making is currently one of the most promising areas of research in the fight against cancer. While the technology is not quite at the stage of using a distributed network of engineered cells, there are already therapies based on altering the decision making of individual cell types.

This is particularly important in the area of adoptive T cell therapy due to the importance of screening desired targets as the adverse effects can lead to the failure of the treatment and, in some cases, death [29]. Thus, the development of methods of designing Boolean circuits plays a key role in the future progress made in immunotherapy as is the case with CAR-T cell therapies. The two important constraints of these

circuits are the number of available logic gates that are implementable in a cell and the types of implementable gates. These constraints present important obstacles to implementing complex logic decisions. Due to the vast number of possible combinations of logic gates, the focus of this work is mainly to look at designing circuits that contain exclusively 2-input NOR (NOR2) gates. NOR2 gates are known as universal logic gates because they can each represent all possible Boolean switching functions. The toxicity increases with an increased number of gates within the same cell, limiting the number of implementable gates within a cell. This limit is oftentimes 7 or less in bacteria and yeast [30, 31, 32], and a lot less in CAR treatments [29]. Thus, even for very simple Boolean operations in a single type of immune cell, it can be necessary to engineer cells with different logic components that communicate together as a part of an implemented distributed computation. Fortunately, cells can communicate through diffusible small molecules (DSMs) which can be transcription factors for bacteria or cytokines for immune cells.

This type of implementation of Boolean functions is made possible by thinking of a Boolean function that maps inputs into outputs in terms of its "disjunctive normal form" (DNF). Each (min)term of this DNF can be implemented in a different cell and then the various outputs of these cells can be combined using a virtual "OR" gate. In practice, various techniques were used to combine multiple minterms to minimize the number of required cells in the design. An "optimal" database (in the sense of using the minimum number of possible NOR2 gates) was used to minimize the number of required gates for each of the components of the overall circuit [1]. This first step of part II focused on circuits requiring 4 inputs or less in an optimal way given biological constraints on the number of gates per cell.

The second half of part II looks at how to deal with circuits that require 5 inputs or more. Because of the NP-hard nature of optimal synthesis algorithm, the commonly used Boolean synthesis algorithms are heuristic in nature, resulting in proposed circuits that are far from optimal. An interesting observation that is reported in this chapter is the fact that the optimal database that was used for 4 inputs or less can be readily used to improve heuristic results that come from widely used tools such as

ABC and SIS [33, 34, 35]. This constitutes an important step forward because such synthesis algorithm shave been fairly stagnant in nature over the past few decades. Finally, an example of a 3-bit adder is shown to demonstrate the ability of the workflow to deal with an implementable problem of a fair complexity. While there is yet to be an example of engineered immunotherapy treatments that make use of 5 or more inputs, the proposed synthesis workflow universally consider all Boolean mappings between inputs and outputs, supporting future experimental work that will eventually require such increased complexity in the decision making of engineered immune cells.

Part I

Chemotherapy and the role of the immune system

Chapter 2

Introduction and Literature Review

Part of this section are reprinted from [20]:

A. P. Tran, M. Ali Al-Radhawi, I. Kareva, J. Wu, D. J. Waxman, and E. D. Sontag, "Delicate balances in cancer chemotherapy: modeling immune recruitment and emergence of systemic drug resistance," Frontiers in Immunology, vol. 11, p. 1376, 2020.

2.1 The role of chemotherapy in the treatment of cancer

Cancer treatments have become more sophisticated and effective based on a better understanding of the underlying biological mechanisms of such treatments on cancerous cells. Available treatments have expanded from traditional surgery, chemotherapy, and radiation therapy to now include a vast array of possibilities including hormone therapy, targeted therapy, stem cell therapy, and immunotherapy. These treatments are most often used together in conjunction to yield a better prognosis. The early treatments were localized in nature with the almost exclusive use of surgery before

1950 and the widespread use of radiation therapy with the discovery of the linear accelerator after 1960. Yet, one of the obvious limitations is in dealing with metastatic tumors that can spread all over the human body [36].

The potential of chemotherapy was first observed in victims of sulfur mustard gas during the First World War when autopsy findings showed the potential of this compound in destroying lymphoid tumors. This convinced Goodman, Gilman, and Lindskog to experiment injecting a similar compound in mice, showing strong tumor regression. This convinced them to attempt it as a treatment of non-Hodgkin's lymphoma, but the effects only lasted a few weeks [37, 38]. Yet, this set a precedent for the use of chemotherapy drugs even though it was not clear why toxins were more prone to killing tumors rather than normal cells. This finding led to other alkylating agents, such as cyclophosphamide and chlorambucil, was found and used to treat patients with lymphomas and leukemias [36]. Very early on, the issue of drug resistance was apparent and remains an important limitation for these treatments.

The finding of Sydney Farber in noticing that folic acid played an important role in treating acute lymphoblastic leukemia (ALL) led to folate analogs such as methotrexate. Methotrexate was used as a treatment for epithelial malignancies, also was shown to be the first drug treatment to cure a solid tumor, and later on, was used as adjuvant therapy following surgery. Combining methotrexate, vincristine, 6-MP, and prednisone in a regimen referred to as POMP, showed to lead to long-term remission in 1965 for children with ALL [39]. In the later parts of the 1960s, the MOPP regimen, made of 4 drugs, showed that it could cure Hodgkin's lymphoma and non-Hodgkin's lymphoma [40, 41]. Employing multiple drugs helps greatly circumvent the problem of drug resistance. In the 1970s and 1980s, the number of drugs discovered were few [36]. The FDA approvals were length, costly, and difficult. The animal models turned out to be unreliable as predictors. On the treatment side, there were severe side effects due to the toxicity of the drugs.

The breakthrough of targeted-therapy came through a better understanding of signaling networks that regulate the proliferation and survival of cellular activities. This also led researchers to find ways to repair defects found in cancer cells lead-

ing to dramatic improvements in the prognosis of treated patients. With this better understanding of cell biology came new targets such as growth factors and signaling molecules [42]. Molecules could be identified through high-throughput screening given desired properties. The approved chemotherapy drugs increased dramatically in the 1990s and beyond [36]. The molecular profiling of human tumors became an essential part of how personal treatments are designed, allowing one to select effective treatments from the constantly evolving array of tools available in the fight against cancer. The use of chemotherapy has changed as treatments are made more targeted but its role is as important as ever. Whether it is used as the main therapy or in an adjuvant role, researchers are exploring the myriads of ways it can effectively be combined with other techniques like immunotherapy, stem cell transplants to restore immune functions, and radiation therapy to increase the overall effectiveness of these combination therapies.

2.2 The importance of drug scheduling in chemotherapy

Immune system involvement in cancer progression has been well established, leading to increased efforts to harness the ability of the host immune system to fight off growing tumors [43]. Standard of care chemotherapy regimens typically involve drug administration on a maximum tolerated dose (MTD) schedule [44]. These regimens aim to target most cancer cells at once, but frequently lead to a reduction in tumor burden only in the short term [18] and often give rise to drug resistance [45, 46]. One reason for this longer-term failure is that MTD-based treatment can cause collateral damage to the host immune system, thus diminishing its ability to target the tumor [18]. An optimal treatment regimen should strike a balance between drug-induced tumor cell kill and damage to the immune system, allowing the two modes of cancer cell elimination to complement each other. Indeed, high frequency low dose drug administration, also known as metronomic chemotherapy, can in some cases strike the

right balance and induce immunogenic cell death (ICD) in tumor tissue by selecting an appropriate choice of drug, dose, and time interval between treatments [47, 48, 14, 49, 50, 51, 3, 15, 52, 16, 17]. Successful achievement of ICD-based therapeutic outcomes during anticancer therapy is dependent on complex interactions between the drug, the tumor, and the host immune system, the nature of which is still being uncovered [2].

To better understand the mechanisms whereby metronomic chemotherapy enables an anti-tumor immune response, it is important to understand how tumors are able to evade immunosurveillance in the first place [43]. An important and well-studied evasion route is through the accumulation of mutations and epigenetic modifications that help avoid immunosurveillance [53, 54].

2.3 Interaction dynamics between tumors and the immune system

During a chemotherapy treatment, the immune system is subjected to combined effects of a variety of immunostimulatory as well as immunosuppressive processes. For background, we next briefly discuss some of these immune-related processes, which involve modifications of the tumor microenvironment (TME). Such modifications include increased acidity resulting from altered nutrient metabolism [55, 56] and altered balance between cytotoxic and regulatory immunity through the recruitment by the tumor of immunosuppressive cells, such as regulatory T cells (Tregs) [57, 58, 59, 60, 61] and myeloid-derived suppressor cells (MDSCs) [62, 63].

Some examples of TME modifications are as follows. Macrophages with an M2 phenotype can produce high levels of TGF-β, IL-10, and vascular endothelial growth factor (VEGF), promoting tumor growth [64, 65, 66, 67]. In other cases, tumor-derived factors and gangliosides can alter dendritic cell (DC) phenotype leading to lower levels of CD80, CD86, CD40 and high indoleamine 2,3-dioxygenase expression that contributes to suppression of T cell immunity [68]. Immunosurveillance can also

be evaded through production of various immunosuppressive cytokines such as TGF-β that play an important role in suppressing macrophages and monocytes [69]. Other factors such as tumor necrosis factor (TNF)-α, IL-1, IL-6, colony stimulating factor (CSF)-1, IL-8, IL-10, and type 1 interferons (INFs) can also contribute to cancer growth [70, 71, 72, 73, 74]. Additionally, pro-angiogenic factors such as VEGF can inhibit differentiation of progenitors into DCs [75]. IL-10 and TGF-β can also inhibit DC maturation. Ganglioside antigens can also suppress cytotoxic T-cells (CTLs) and dendritic cells (DCs) function [76]. Immunosuppressive enzymes such as IDO, arginase, and inhibitor of nuclear factor kappa-B kinase (IKK)2 may also contribute to tumor progression via direct actions on tumor cell proliferation or through induction of T cell tolerance/suppression [77, 78, 79].

2.4 Network effects of chemotherapy interventions

By targeting various components that regulate immune tolerance, cancer chemotherapy drugs, such as mitoxantrone, idarubicin, doxorubicin, and cyclophosphamide can induce immunogenic cancer cell death in addition to their direct cancer cell cytotoxic effects [80, 51, 50, 81]. By using an optimized drug dose and schedule of administration, favorable immune responses can be achieved, including increases in macrophage recruitment and maturation [82], proliferation of NK cells, levels of IFN-γ [83], as well as elevated post-apoptotic release of the nuclear chromatin binding protein HMGB1, which can stimulate antigen presentation by DCs, helping $CD8^+$ T cell activation [84, 85]. In some cases, type-1 interferon signaling pathways are switched on, leading to host antitumor immunity activation [86, 87, 88, 25]. Immunosuppressive molecules, like CD31, CD46, CD47, are downregulated on dendritic cells by ICD drug treatment [89]. Molecular chaperones such as HSP90 appear on the tumor cell surface, promoting DC maturation [90]. These optimized drug administration conditions can also lead to transient lymphopenia, which upregulates repair mechanisms and can lead to a vast array of immunostimulatory outcomes, including enhanced T-cell activation, immune recruitment, DC differentiation and maturation, as well as the release of large

amounts of chemokines and cytokines [91, 92]. Cytotoxic effects on immunoregulatory cells, such as MDSCs and Tregs, contribute to restoration of anti-tumor immunity by decreasing suppression of T-cells and NK cells [93]. Other factors affecting immunogenicity include changes in MHC-1 molecules and tumor-specific antigens on the tumor cell surface [94], stress-induced expression of NK cell stimulatory ligands, and decreases in NK inhibitory ligands [95].

The ability of drugs to induce anti-tumor immune responses is not sufficient by itself to ensure a successful therapeutic response, as the effect on various compartments of the immune system, and thus on overall tumor burden can vary dramatically depending on dose, schedule and tumor type. Scheduling and dosing of an ICD drug is of critical importance in instigating an immune response, which relates to the concept of "getting things just right" [2, 19]. For instance, administration of cyclophosphamide on a 6-day repeating schedule (Q6D) at 140 mg/kg per dose, dramatically improves the therapeutic outcome for murine GL261 gliomas through immunomodulatory mechanisms [3, 15, 16]. Other drug treatment schedules, however, do not result in the same efficacy. This loss of efficacy correlated with reversal of an initial anti-tumor immune response, despite ongoing ICD drug treatments [3]. Intriguingly, cyclophosphamide treatment of Lewis lung carcinoma (LLC) and B16F10 tumor at the same dose and on the same Q6D schedule does not result in tumor regression or immune cell recruitment, despite the intrinsic sensitivity of these two tumor lines to activated cyclophosphamide [17].

It is clear that much remains to understand about the underlying mechanisms of ICD action including the impact of chemotherapy dose and schedule on the many factors linked to the ICD response [2]. Well-designed mathematical models, which can help elucidate the complex interplay between the various players, make these models an invaluable complementary tool to *in vivo* experimental results for designing better treatments.

2.5 Mathematical modeling of tumors and the immune system

There is a rich tradition in utilizing mathematical biology to study cancer chemotherapy and the immune system. A large variety of models have been proposed, notably in the work of de Pillis and collaborators, who introduced a series of models that depict many of the interactions between chemotherapy and immunotherapy drugs, the immune system, and tumor progression [96, 97]. Closely related to our topic, Ciccolini et al. [23] proposed a pharmacokinetics and pharmacodynamics (PKPD) model for metronomic chemotherapy using gemcitabine, one that considers the effects of cytotoxicity on endothelial cells and the emergence of drug resistance. Ledzewicz, Behrooz, and Schättler proposed a minimally parameterized mathematical model for low-dose metronomic chemotherapy that explicitly considers tumor vasculature [22], and in subsequent work [98] applied optimal control theory to this system, so as to devise a treatment schedule that can minimize tumor burden subject to appropriate constraints. To the best of our knowledge, only one study has looked at modeling the immune recruitment by ICD drugs [19]. In that work, however, there was no experimental validation of the proposed model.

Numerous cancer models have been proposed to account for the emergence of therapeutic resistance due to cancer cell heterogeneity. See, for example, the extensive references in [24]. However, to our knowledge, no previous work has systematically and theoretically modeled what we call "systemic drug resistance," by which we mean resistance as an immune-mediated dynamical phenomenon.

Chapter 3

Delicate Balances in Cancer Chemotherapy: Modeling Immune Recruitment and Emergence of Systemic Drug Resistance

Reprinted from [20]:

A. P. Tran, M. Ali Al-Radhawi, I. Kareva, J. Wu, D. J. Waxman, and E. D. Sontag, "Delicate balances in cancer chemotherapy: modeling immune recruitment and emergence of systemic drug resistance," Frontiers in Immunology, vol. 11, p. 1376, 2020.

3.1 Motivation of this work

Here, we propose a mathematical model that is fit to experimentally observed tumor growth curves in GL261 tumor-bearing SCID mice that were given metronomic chemotherapy at Q6D to Q12D drug administration regimens [3]. The proposed model incorporates immune cell recruitment, as well as pharmacokinetics of cyclophosphamide. First, we find a fixed set of model parameters, not adjusted for

individuals or for drug schedule, that fit experimental data across these various drug regimens very well. To further validate the model, we not only compare the experimental fits to the measured tumor volume data, but also ask if the "latent variables" in our model, which represent immune activation, can reproduce a second set of experimental data, not used in training the model. Specifically, we asked if peak immune activation times in our model correspond to experimentally measured times. Finally, we investigate how our validated model can be used as a tool to identify better treatment schedules, to build a quantitative understanding of the mutual interplay between drug, tumor, and immune system and to better predict drugs that induce immune cell recruitment in a clinical setting.

3.2 Development of the mathematical model

3.2.1 Background on the experimental results using cyclophosphamide

Summary of experimental data

The experimental data used in developing our mathematical model were derived from previous work by Wu and Waxman [3], where cultured GL261 gliomas cells were implanted in SCID mice. Tumors were allowed to grow to 300 to 1000 mm^3, at which point the mice were treated with cyclophosphamide (CPA), given on different metronomic schedules. Greatest tumor burden reduction was observed when repeated doses of 140 mg CPA/kg-BW (body weight) were administered every 6 days. Comparisons were made between the every 6-day schedule (Q6D) and three other schedules: treatment every 9 days (Q9D); treatment alternating between every 6 days and every 9 days; and treatment every 12 days (Q12D). Additionally, a dose of 210 mg CPA/kg-BW was given every 9 days, ensuring the mice receive the same total amount of drug as when 140 mg/kg doses were given on a Q6D schedule, to evaluate the impact of schedule vs. total dose on the final outcome. Tumor growth curves were reported for drug-free controls, and for the following regimens: a single CPA administration given

on day 0 (1-CPA), two CPA treatments given on days 0 and 6 (2-CPA), as well as three CPA treatments given on days 0, 6, and 12 (3-CPA). This previous work also reported relative gene expression levels for immune cell markers for NK cells, dendritic cells, and macrophages for the 1-CPA, 2-CPA and 3-CPA treatment regimens. There were $n = 4 - 12$ tumors per treatment group.

Published data [3] suggest that the host immune system takes between 6 and 12 days to start significantly impacting the tumor growth. From this, the authors infer that any initial slowdown in tumor growth, within 1 or 2 days of drug injection, is likely caused by cyclophosphamide-induced tumor cell death. A decrease in immune cell number immediately after drug administration was observed, highlighting the drug's cytotoxic effect on immune cells as well as cancer cells. Notably, the chemoattractant CXCL10, which acts on many innate immune cells, and which is induced by IFN-λ, increased following the first CPA injection, peaking around six days post administration. Between six and twelve days post injection, there was also an increase in other innate immune cells markers, such as Nkp46, Nkg2d, Prf1, Gzmb for NK cells, Cd207 and Cd74 for dendritic cells, and Cd68 and Emr1 for macrophages [3].

Based on the impact of various treatment schedules tested on tumor volume, it is apparent that 12-18 days after CPA treatment is halted, the GL261 tumors cease to shrink and then start to regrow. Based on treatments where the same dose of CPA is repeated at regular intervals, the authors concluded that a 6-day break between CPA doses is ideal for maintaining prolonged immune cell recruitment as well as tumor shrinkage. Increasing the number of drug-free days between treatments from 6 to 9 to 12 days increased the number of tumors that circumvented the therapeutic effects of the drug, resulting in a shorter interval prior to tumor regrowth [15]. Further analysis of tumor infiltrating immune cells in these models indicated there is a strong correlation between loss of immune response and tumor escape. Another study indicated that breaks in drug treatment shorter than 6 days also resulted in worse performance for the same AUC (area under the curve for the administration of the drug) with noted absence in immune recruitment if CPA was given to the mice daily [15]. One of the first goals of our model is to recapitulate these results.

Modeling the immune-tumor-drug interactions

A schematic of our model, highlighting the main interactions between tumor, drug, and the immune system, is shown in Fig. 3.1(a).

Cyclophosphamide can be toxic to the immune system, as confirmed by experimental data showing that, one day after cyclophosphamide administration, there is a significant decrease in marker gene expression for NK cells, DCs, and macrophages in the tumor microenvironment [3]. High doses of cytotoxic drugs that damage immune cells can diminish or even nullify their ability to target the tumor. Given this finding, it is not surprising that traditional MTD chemotherapy not only has substantial side effects on the patient's health, but also leads to immunosuppression and increases the risk of tumor relapse due to drug resistance. This then naturally leads to the question of how to represent (in a concise but not oversimplified mathematical way) the immunostimulatory and immunosuppressive effects of drug treatment when using a metronomic regimen of an ICD drug in order to eventually find a way to balance out these two forces.

The paradoxical effect in which a drug reduces immune cell counts in the short term, but also enhances the immune system in the longer term, is an instance of an "incoherent feed-forward loop" (IFFL). A similar paradoxical effect is that of the effect of treatment on cancer growth: on the one hand, the drug directly attacks the tumor, but on the other hand, through "friendly fire" also attenuates immune activity, thus degrading the anti-tumor response. IFFLs constitute one of the core network motifs in systems biology [21], and are found in processes as varied as gene regulation, immune recognition, synthetic biology, biological sensors, and bacterial motion [99, 100, 101, 102, 103, 26, 104, 105, 106].

Some of the most common forms of IFFLs are illustrated by IFFL-I and IFFL-II block diagrams in Fig. 3.2.

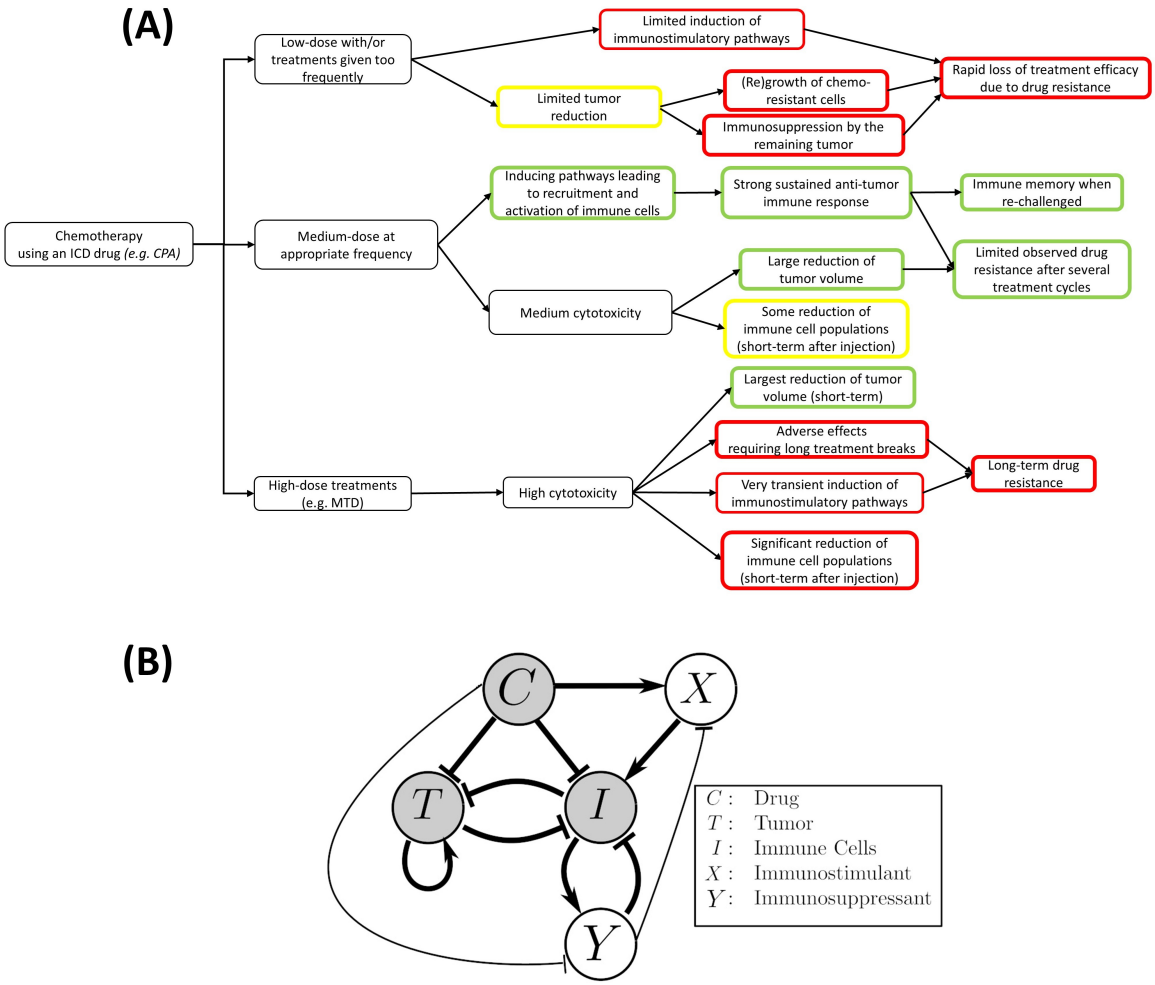


Figure 3.1: The focal point of this work is to improve our mechanistic understanding on how application the right metronomic chemotherapy regimen can, in specific cases, stimulate a large increase in the immune response. Both the immunosuppressive and immunostimulatory effects of the drug are considered in our model, with the coexistence of these two seemingly opposite effects being an important point of emphasis. (A) An outline of the effects of different chemotherapy treatments on the immune system and tumor growth as observed in the context of an ICD drug like cyclophosphamide [2]. (B) The translation of the outlined experimental observations into a detailed schematic diagram of the interactions that formed the basis of our model. Thicker arrows represent well-known effects. Thinner arrows are hypothesized to be present, and also arise from our numerical fits to the data. Notably, we simplified the system such that the interactions between tumor and immune cells leading to increased immune activity are omitted in the context of this work. The details about this assumption can be found in the Methods section.

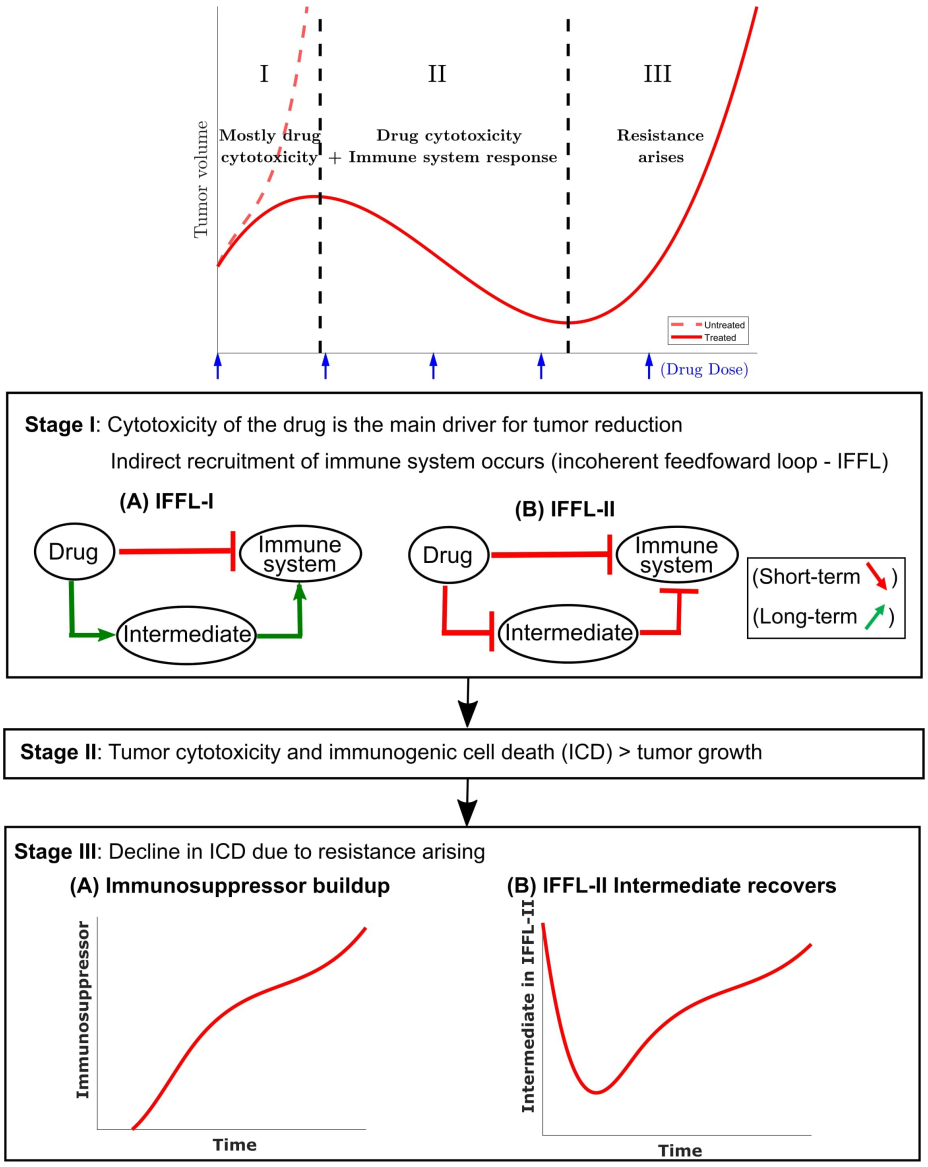


Figure 3.2: A conceptual tumor growth curve under metronomic chemotherapy treatment with repeated doses is shown and is broken down into three representative stages. In stage I, tumor growth is primarily inhibited by the direct tumor cell cytotoxicity of the drug. There is typically a delay in the immune response due to the immune cell cytotoxicity of the injected drug. The immune cell data in [3] indicate that an immune response is substantially reduced immediately after drug administration, but eventually recovers and leads to a strong response 6 to 12 days later. Two possible incoherent feed-forward loop (IFFL) motifs that capture these short-term and long-term dynamics of the drug-immune system interactions are shown. In stage II, the tumor volume is typically reduced as the induced immune response becomes the principal contributor to tumor cell death. Stage II lasts until the induced immune response fades due to the emergence of drug resistance. Stage III typically has the tumor recovering its ability to grow in an exponential fashion. Two possible scenarios are shown with the occurrence of an immunosuppressor build up or recovery of the intermediate in an IFFL-II scenario.

In our context, two forms of IFFL, promotion of an immunostimulatory intermediate (IFFL-I) and "repressing the repressor" (IFFL-II), are both likely scenarios, and could occur in conjunction. Accordingly, to represent the immune system recruitment behavior, the mathematical model must include an intermediate that leads to immune cell recruitment in the longer term. The possible mechanisms are numerous, as summarized in the Introduction, and it is likely that the observed tumor growth is the result of both a repressor being repressed and some immunostimulatory element. Due to the lack of extensive immune data measurements, we only considered the immunostimulatory pathway in this work.

In the event of tumor escape depicted conceptually in the bottom section of Fig. 3.2, the experimental data in [3] suggests that the immune system, which was reinforced through the first few injections of cyclophosphamide, fails to maintain a level of immune cell recruitment sufficient to allow the host to keep the tumor at bay.

3.2.2 Mathematical model

The full mathematical model is given by a coupled system of five ordinary differential equations (ODEs). The first equation describes the change in CPA concentration over time (drug pharmacokinetics):

$$\frac{dC}{dt} = u - \frac{k_1 C}{k_2 + C} \tag{3.1}$$

and the remaining four describe the effect of CPA on cancer and immune cells (pharmacodynamics):

$$\frac{dT}{dt} = k_a T - \frac{k_b C T}{k_c + T} - k_d T I \tag{3.2}$$

$$\frac{dI}{dt} = X - k_e T I - k_f C I - k_g Y I - k_h I \tag{3.3}$$

$$\frac{dX}{dt} = \frac{C}{1 + C/k_i} - k_j X - k_k X Y \tag{3.4}$$

$$\frac{dY}{dt} = \frac{I}{1 + C/k_l} - k_m Y C \tag{3.5}$$

In the next section we describe in detail the various terms of the model, including the role of the variables $C = C(t), \ldots, Y = Y(t)$, which represent time-dependent changes of drug, tumor, and immune components, as well as provide descriptions and interpretation of the parameters $k_1, \ldots k_n$. The input variable $u = u(t)$ is introduced to describe drug administration. The equation terms with missing parameters are the result of a nondimensionalization step, the details of which are introduced a little bit later in this section. It facilitates model analysis without loss of utility.

The PK submodel

Equation (1) describes the change over time in concentration of cyclophosphamide $C(t)$ in the tumor microenvironment. It is assumed that the drug is administered at a time-dependent rate $u(t)$ and is cleared at a Michaelis-Menten (saturated) rate $\frac{k_1 C}{k_2 + C}$. This compartment is assumed to be where the interactions between the drug and its targets are assumed to take place.

Parameters k_1 and k_2 are obtained as part of the global fit to experimental data described later.

Our simple PK model is phenomenological and is intended to capture the delay in drug activity with respect to various cell types rather than all the details of CPA activity, which is sufficient for the purposes of our investigation. Furthermore, although the steps of CPA activation by liver cytochrome P450 metabolism have been well studied, including the pharmacokinetics of CPA in mice [107, 108], details of how the intermediates interact in real time with the immune system and the tumor cells remain largely unknown. In using a one-compartment model, the fast dynamics of the drug reaching the blood stream and the decomposition of CPA into its metabolites are assumed to have occurred (as captured by the term $u(t)$) prior to the ability of the drug to impact system dynamics, reflecting the difference in time scales of drug PK and tumor-immune dynamics.

Tumor and immune dynamics

The full model describes interactions between five variables. These represent the tumor volume, denoted by $T(t)$, the immune response denoted by $I(t)$, and two phenomenological variables: an immunostimulatory intermediate species and an immunosuppressive intermediate species, whose counts are denoted by $X(t)$ and $Y(t)$, respectively. All these variables except $T(t)$ are phenomenological representations of complex underlying phenomena, lumping together both cellular populations and chemical signals such as cytokines, and thus do not carry any biologically meaningful units. Such an approach allows identifying broad functional classes of actors that impact the observed dynamics, which can then be teased out in more detail in future investigations. All three of these species are directly affected by the concentration $C(t)$ of cytotoxic drug.

We assume that tumor volume grows exponentially, at a rate k_a. In contrast to other mathematical models, we do not introduce saturation or crowding terms for tumor growth, because the data used in fitting is from mouse models, where a maximum tumor volume (or carrying capacity) is never reached for humane reasons. In our model, tumor cells can be killed by two different mechanisms: either by interaction with the drug $C(t)$ at a rate k_b, or by immune cells $I(t)$ at a rate k_d. The ratio $\frac{T}{k_c+T}$ captures the fact that the cytotoxic death term is proportional to T when the tumor is small but saturates when T becomes much larger than k_c.

Taking into account these mechanisms results in the following equation for change in tumor size over time:

$$\frac{dT}{dt} = \underbrace{k_a T}_{\text{exponential growth}} - \underbrace{\frac{k_b T C}{k_c + T}}_{\text{cytotoxic death}} - \underbrace{k_d T I}_{\text{death as a result of immune-tumor interactions}} . \quad (3.6)$$

For the purposes of this analysis, we do not differentiate between various types of cytotoxic immune cells nor do we distinguish the effects of these cells from other immune factors such as chemokines or cytokines; instead, we track the change over time of an aggregate immune indicator $I(t)$. We assume that this indicator increases

through direct interaction with immunostimulatory intermediate $X(t)$ (which will be described next), and can either be inactivated through interactions with tumor cells at a rate k_e, can decrease ("or die if seen as immune cells") due to exposure to drug $C(t)$ at a rate k_f, can become suppressed through interaction with immune suppressor $Y(t)$ at a rate k_g, and can decrease at a natural rate k_h. The term $-k_eTI$ represents both the activation $k_{e+}TI$ and inactivation of the immune response by tumor cells $-k_{e-}TI$ with the assumption that $-k_eTI = k_{e+}TI - k_{e-}TI$. While fitting these parameters to data, we found that defining the tumor-immune interaction term in this particular functional form, i.e., $-k_eTI$, led to a positive k_e value. This observation was likely due to the data being collected at a late stage in tumor growth progression. While one may expect the activation and inactivation phenomena to have different functional forms, the simplification was also made due to the experimental data suggesting that the recruitment by a $k_{e+}TI$ term on the immune cells, cytokines and chemokines is smaller by at least an order of magnitude when compared to the immune recruitment driven by the metronomic chemotherapy treatment.

This results in simplification of the following equation for change in the indicator I over time from:

$$\frac{dI}{dt} = x_1 - (k_{e-} - k_{e+})\,TI - k_f CI - k_g YI - k_h I\,, \tag{3.7}$$

to:

$$\frac{dI}{dt} = \underbrace{X_1}_{\text{drug-mediated immune recruitment}} - \underbrace{k_eTI}_{\text{exhaustion}} - \underbrace{k_fCI}_{\text{death by drug}} - \underbrace{k_gYI}_{\text{immunosupression}} - \underbrace{k_hI}_{\text{decay}}\,. \tag{3.8}$$

Next, in addition to tracking the dynamics of tumor and immune cells, we introduce two phenomenological variables that are both affected by the drug, and can in turn affect both tumor and immune cells.

Firstly, we introduce an immunostimulatory intermediate $X(t)$, which impacts immune cell recruitment. We assume that it can be increased by an ICD type drug,

such as cyclophosphamide, according to saturating function $\frac{C}{1+C/k_i}$, where immune cell recruitment is linear up to a threshold k_k, and becomes saturated when $C(t) > k_i$, representing an upper bound of drug-induced immune recruitment. The immunostimulatory intermediate $X(t)$ is assumed to decay at a rate k_j, and to be inactivated through interactions with an immunosuppressive factor $Y(t)$, which will be defined next. The resulting equation for change over time of immunostimulatory factor $X(t)$ is as follows:

$$\frac{dX}{dt} = \underbrace{\frac{C}{1+C/k_i}}_{\text{recruitment by drug}} - \underbrace{k_j X}_{\text{decay}} - \underbrace{k_k XY}_{\text{immunosupression}} . \tag{3.9}$$

Finally, we introduce the immunosuppressive factor $Y(t)$, which can impact tumor-immune dynamics and which in itself is affected by the drug $C(t)$. Its dynamics over time are described by the following equation:

$$\frac{dY}{dt} = \frac{I}{1+C/k_l} - k_m Y C. \tag{3.10}$$

The immunosuppressive intermediate $Y(t)$ is assumed to be induced by I; its effectiveness can be affected by drug $C(t)$, which is captured as $\frac{1}{1+C/k_l}$. It can also be cleared through interaction with the drug $C(t)$ at a rate k_m.

The model accounts for cytotoxic effects of cyclophosphamide, not only on cytotoxic immune cells, such as NK and CD8+T cells, but also on immunosuppressive cells, such as MDSCs and Tregs [109, 110, 95]. The immunosuppressive intermediate Y is vital to the appearance of systemic drug resistance when the treatments are continuously repeated over long treatment periods.

A schematic summary of these interactions is shown in Fig. 3.1(b). A structural identifiability analysis was performed using [111] and showed that all parameters are (locally) identifiable.

3.2.3 Nondimensionalization of the model equations

Consider the equations of the model:

$$\frac{dC}{dt} = u - \frac{k_1 C}{k_2 + C} \tag{3.11}$$

$$\frac{dT}{dt} = k_{a'} T - \frac{k_{b'} CT}{k_{c'} + T} - k_{d'} TI \tag{3.12}$$

$$\frac{dI}{dt} = k_{e'} X - k_{f'} TI - k_{g'} CI - k_{h'} IY - k_{i'} I \tag{3.13}$$

$$\frac{dX}{dt} = \frac{k_{j'} C}{1 + C/k_{k'}} - k_{l'} X - k_{m'} XY \tag{3.14}$$

$$\frac{dY}{dt} = \frac{k_{n'} I}{1 + C/k_{o'}} - k_{p'} YC \tag{3.15}$$

$$\tag{3.16}$$

Substituting I, X, and Y using the relationships:

$$\left[I = I^* \hat{I}, X = X^* \hat{X}, Y = Y^* \hat{Y} \right]$$

and limiting the scope to the equations for these state variables to be nondimensionalized yield the following set of equations:

$$\frac{d\hat{I}}{dt} = \frac{k_{e'} X_1^* \hat{X}_1}{I^*} - k_{f'} T\hat{I} - k_{g'} C\hat{I} - k_{h'} \hat{I} Y^* \hat{Y} - k_{i'} \hat{I} \tag{3.17}$$

$$\frac{d\hat{X}}{dt} = \frac{k_{j'} C}{X^* (1 + C/k_{k'})} - k_{l'} \hat{X} - k_{m'} \hat{X} Y^* \hat{Y} \tag{3.18}$$

$$\frac{d\hat{Y}}{dt} = \frac{k_{n'} I^* \hat{I}}{Y^* (1 + C/k_{o'})} - k_{p'} \hat{Y} C. \tag{3.19}$$

Making the following replacements

$$[I^* = k_{e'} k_{j'}, Y^* = k_{e'} k_{j'} k_{n'}, X^* = k_{j'}]$$

and rewriting the parameter names yield the following nondimensionalized set of equations with 4 less parameters:

$$\frac{dC}{dt} = u - \frac{k_1 C}{k_2 + C} \tag{3.20}$$

$$\frac{dT}{dt} = k_a T - \frac{k_b C T}{k_c + T} - k_d T I \tag{3.21}$$

$$\frac{dI}{dt} = X - k_e T I - k_f C I - k_g I Y - k_h I \tag{3.22}$$

$$\frac{dX}{dt} = \frac{C}{1 + C/k_i} - k_j X - k_k X Y \tag{3.23}$$

$$\frac{dY}{dt} = \frac{I}{1 + C/k_l} - k_m Y C. \tag{3.24}$$

3.2.4 Fitting methodology

Error criterion

In this work, an objective function was defined for the nonlinear optimization problem that is used to fit the model parameters:

$$\underset{\hat{y_i}}{\operatorname{argmin}} \sum_{i=1}^{N} \operatorname{sat}\left[\left(\frac{y_i + a}{y_i + b}\right)\left(\frac{|y_i - \hat{y_i}|}{y_i + c}\right)\right] \tag{3.25}$$

with y_i being the experimental value and $\hat{y_i}$ the predicted value. $\operatorname{sat}(x)$ is a saturation function such that $\operatorname{sat}(x) = 1$ for values of $x > 1$. The values of a, b, and c are determined in such a way that low values of y_i are not weighted too strongly. In the objective function, b and c can be interchangeable, so we will assume that $b < c$. In the limit that $b < y_i \ll a, c$, we get that:

$$\underset{\hat{y_i}}{\operatorname{argmin}} \sum_{i=1}^{N} \operatorname{sat}\left(\frac{a}{c}\frac{|y_i - \hat{y_i}|}{y_i + b}\right) \tag{3.26}$$

a and c are chosen such that $\frac{a}{c} < 1$ and b is small value that plays both the role of a regularizing term and avoids a division by 0. In the limit when y_i is large, we get:

$$\underset{\hat{y}_i}{\operatorname{argmin}} \sum_{i=1}^{N} \operatorname{sat}\left(\frac{|y_i - \hat{y}_i|}{y_i}\right) \tag{3.27}$$

which is a standard normalization. The magnitudes of a and c play an important role in how quickly this limit of large y_i is approached. Let's note that if $y_i \ll b \ll a, c$ then the ratio becomes $\frac{a}{c}\frac{|y_i - \hat{y_i}|}{b}$. The parameters are chosen such that $bc \gg a$, small values are filtered out, as these may be more susceptible to measurement noise or below a threshold of detection.

The values of a, b, and c in the error criterion were chosen as 19.31, 1, and 227.6, respectively. These values were found to provide a balance between providing enough weights to errors involving small experimental values, while also ensuring that the desired phenomena of tumor evasion and immune recruitment are captured appropriately by the model.

Determination of the population fits

The main underlying assumption of the population fits is that all mice are characterized by the same parameters. In reality, there can be a myriad of ways mice can differ from one another. Notably, their immune system might be of different strength when fighting the tumor and different tumors can grow at different speeds. However, the immune data that was available [3] is confined to the 1-CPA, 2-CPA, and 3-CPA scenarios and only three individual mice per experiment. The tumor growth curves were thus the only data that was used to fit the model.

In addition to having the same parameters, the initial values of all state variables were assumed to be 0, except for the initial tumor volume. While the mice do have a still functioning immune system at the moment they are being monitored, an escaping tumor is a sign that the immune system is compromised or unable to contain the tumor growth. Thus, the underlying assumption is that the immune system at the start of treatment is either of negligible effect or its effect on the tumor growth is

lumped inside the k_a constant. Furthermore, the immune data from [3], also shown in Fig. 7, shows that the effect of cyclophosphamide given on a metronomic regimen can yield an order of magnitude increase in the gene expression of innate cell immune markers, and similar observations can be seen for the gene expression of other markers for cytokines, chemokines, and adhesion molecules.

The last assumption for the population parameters is that the input u appearing in Eq. 1 consists of a step input of 140 mg/kg in concentration, applied at the moment a dose injection. From the time-scale of the experiments, the time-scale of the drug injection is negligible when monitoring the tumor size every few days. The input and the state variables besides the tumor (T) are assumed to be unitless. For the drug regimen of CPA/9-days(210mg/kg), the step input for u was increased to 210 mg/kg in concentration for each dose injection in order to account for the higher dose.

Handling the outliers

In this data, there are two types of outliers that were taken out of the analysis:

- Time series with average tumor volume above a given cutoff value of 2500 mm^3 and that did not belong to the untreated group.
- Time series that contradicted the behavior of neighboring curves.

The first criterion led to the exclusion of 3 outliers out of 65 time series. Large tumors followed dynamics different from what the model could explain. However, the large tumor data is very sparse, so there did not seem to be enough data to confidently elucidate the functional form that governs these data points.

Given the interest in finding a set of population parameters that captured the quantitative and qualitative behaviors of the experimental data, a second criterion was used to avoid fitting the model with contradictory behaviors. To quantitatively measure these deviations, the experimental values of the tumor volumes at each time point at a given treatment condition are ranked. Defining r_{ijk} that represents the rank for the time point i of time series experiment j at the treatment condition k.

This rank can be used to form a rank vector R_{jk} such that:

$$R_{jk} = \{r_{1jk}, ..., r_{Njk}\} \text{ for } N \text{ time points.} \tag{3.28}$$

For each time series, a scalar D_{jk} is calculated using the following formula:

$$D_{jk} = \frac{\text{std}\,(R_{jk})}{M_k} \tag{3.29}$$

with std representing the calculation of a standard deviation and M_k the number of time series data at a given treatment condition k. When two or more tumors at a given time point and treatment condition had the same recorded tumor volume, the average rank of these tumors was assigned for r_{ijk}.

The outliers in the data are shown in Fig. 3.3 and were picked if they were singled out by the first criterion and/or second criterion. The exclusion rule for the second criterion was defined as $D_{jk} > 0.21$.

3.3 Results

In this section, we first use the phenomenological model described above to fit experimentally observed tumor growth curves. We then validate the model by comparing model predictions for the immune compartment to experimental data; notably, the data for the immune compartment were not fit, and thus provide an independent validation of the model, where a single set of parameter values was sufficient to recapitulate experimentally observed dynamics. Finally, the validated model is used to make predictions about treatment regimens that have not been tested experimentally.

Here, the notations 1-CPA, 2-CPA, 3-CPA are used to indicate 1, 2 or 3 doses of CPA that are given 6 days apart. The first dose is always given on day 0. CPA/6-days, CPA/9-days, CPA/12-days indicate that treatments were given 6, 9, or 12 days apart, respectively. CPA/9days(210mg/kg) indicates that the drug doses of 210 mg/kg (rather than 140 mg/kg in other treatment groups) were administered 9 days apart. Finally, the abbreviation CPA/6-9days indicates a break of 6 days between

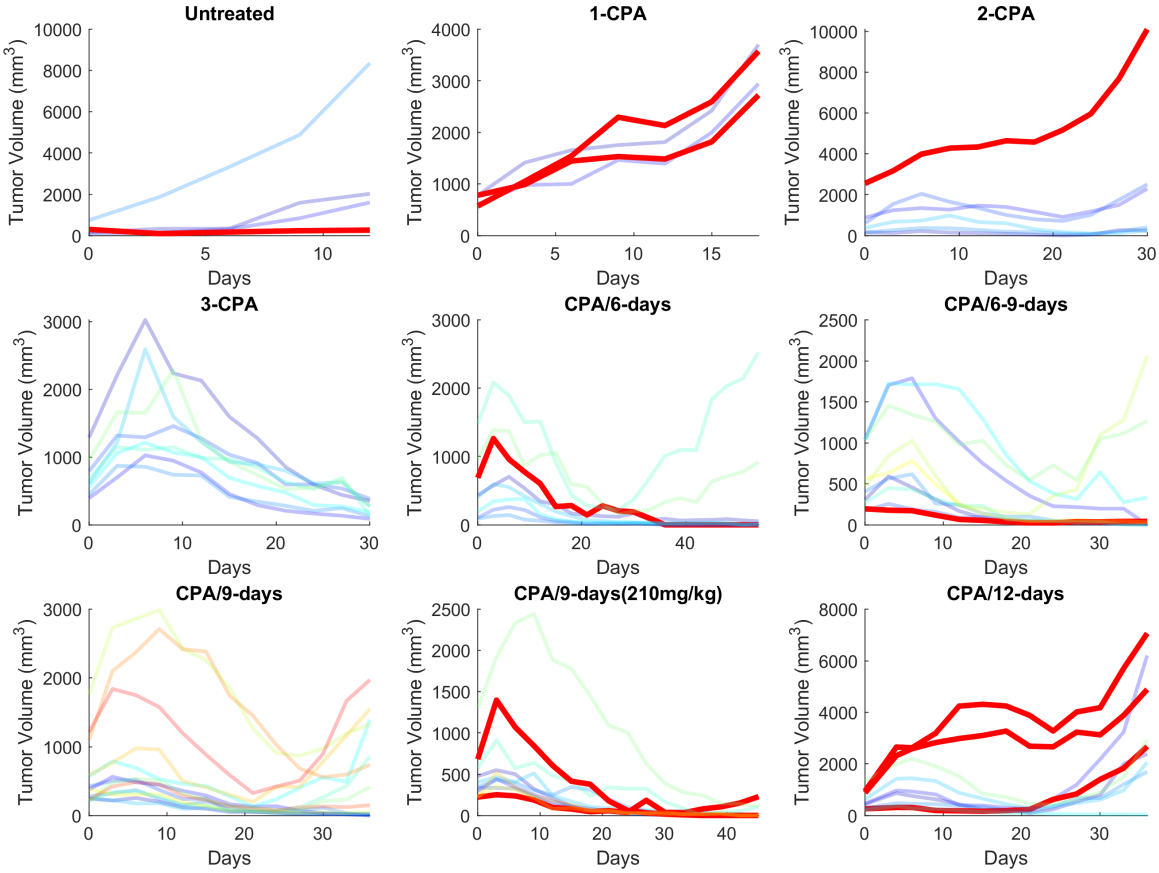


Figure 3.3: The experimental data in [3] are plotted with the outliers highlighted in red. Out of the 65 time series data considering 9 different treatment conditions, 11 were excluded in the fitting of the population parameters.

Table 3.1: Parameter values for fits within 1% of optimal value found for the objective function.

Param.	**Values**						
	Fit A	Fit B	Fit C	Fit D	Fit E	Fit F	Fit G
k_a	0.2211	0.2060	0.2181	0.2180	0.2019	0.2292	0.2164
k_b	0.1035	0.07393	0.1674	0.06062	0.1001	0.1995	0.6627
k_c	154.8	139.7	127.7	127.4	97.83	294.7	1066
k_d	332.3	574.6	4415	809.1	4543	1.629×10^4	3723
k_e	1.042	5.621	0.3411	14.08	2.026	15.73	0.1380
k_f	20.42	0.3397	70.21	6.443	4.312	273.9	184.3
k_g	1.743×10^6	6.008×10^5	4.172×10^7	4.147×10^7	2.623×10^6	6.281×10^4	3.106×10^7
k_h	3.443×10^4	1.883×10^5	7.195×10^4	4.712×10^5	4.900×10^4	1.705×10^6	2.269×10^5
k_i	5.435	15.84	0.7552	36.49	0.4293	5.545	2.458
k_j	1.442×10^{-4}	5.571×10^{-3}	6.979×10^{-4}	1.518×10^{-4}	1.528×10^{-4}	0.04704	1.442×10^{-4}
k_k	9.413	31.61	156.0	25.73	223.7	679.0	157.5
k_l	585.2	461.0	233.8	6516	438.1	612.2	821.8
k_m	1.016×10^{-6}	1.015×10^{-6}	1.053×10^{-6}	1.018×10^{-6}	1.022×10^{-6}	1.027×10^{-6}	1.016×10^{-6}
k_1	41.02	62.28	192.8	52.33	50.08	530.5	40.43
k_2	142.0	400.4	747.2	454.4	145.5	3450	148.6

first and second doses, and a break of 9 days between second and third doses.

3.3.1 Population fits to the experimental data

The experimental growth curves for individual tumors were obtained from [3] and fitted using the aforementioned objective function. The minimization of the error criterion was carried out using *fmincon* with the interior-point algorithm in MATLAB (Release R2019a, Mathworks, MA).

The initial values of the state variables were assumed to be 0 except for tumor volume. The initial guesses to the optimization problem used for finding population parameter fits were drawn from a uniform distribution on a log-scale to sample several orders of magnitude. Numerous initial guesses ($> 10^5$) were tested using the Northeastern Discovery computer cluster. Due to the nonlinearity of the model, multiple sets of parameters yielded equally good minimal objective function evaluations. Out of 16,000 starting points for the optimization, the 8 fits within 1% of the objective function optimal value are summarized in Table 3.1. The excluded outliers are highlighted in Fig. 3.3.

In Figs. 3.4 and 3.5, the simulated growth curves generated using parameters summarized in Fit A in Table 3.1 are shown side by side with the corresponding experimental data. This Fit A was picked among the 8 best fits due to it better capturing the qualitative behavior for CPA/12days, but the differences between these fits among the other sets of treatment conditions were relatively small. Additionally, the predicted pharmacokinetics are shown in Fig. 3.9. It is important to note that each treatment condition corresponds to an experimental dataset. Consequently, variations between datasets can be more prominent than within the same dataset. Also, the units were not emphasized due to three out of the five state variables being dimensionless. However, it is straightforward to adapt the proposed model in the presence of such experimental data by applying the nondimensionalization transformations introduced earlier and obtain the appropriate units.

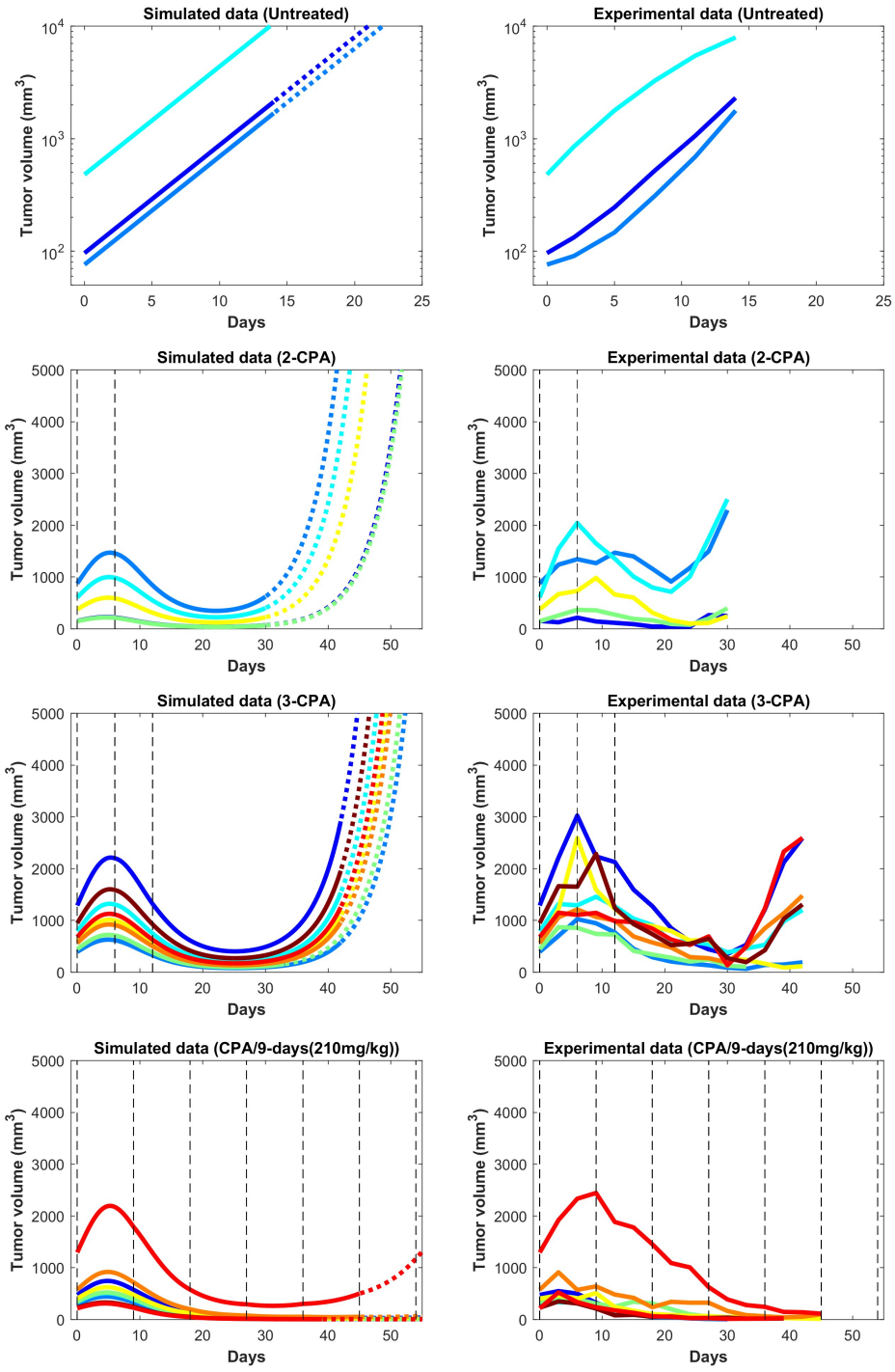


Figure 3.4: Simulated and experimental growth curves for the scenarios of 1-CPA, 2-CPA, 3-CPA and doses of 210 mg/kg given every 9 days (CPA/9-days/210). When not specified, CPA doses are 140 mg/kg. Black dashed lines represent the time at which drug treatments are given. Solid lines represent fitted data; dotted lines represent predicted growth curves extrapolated from the model.

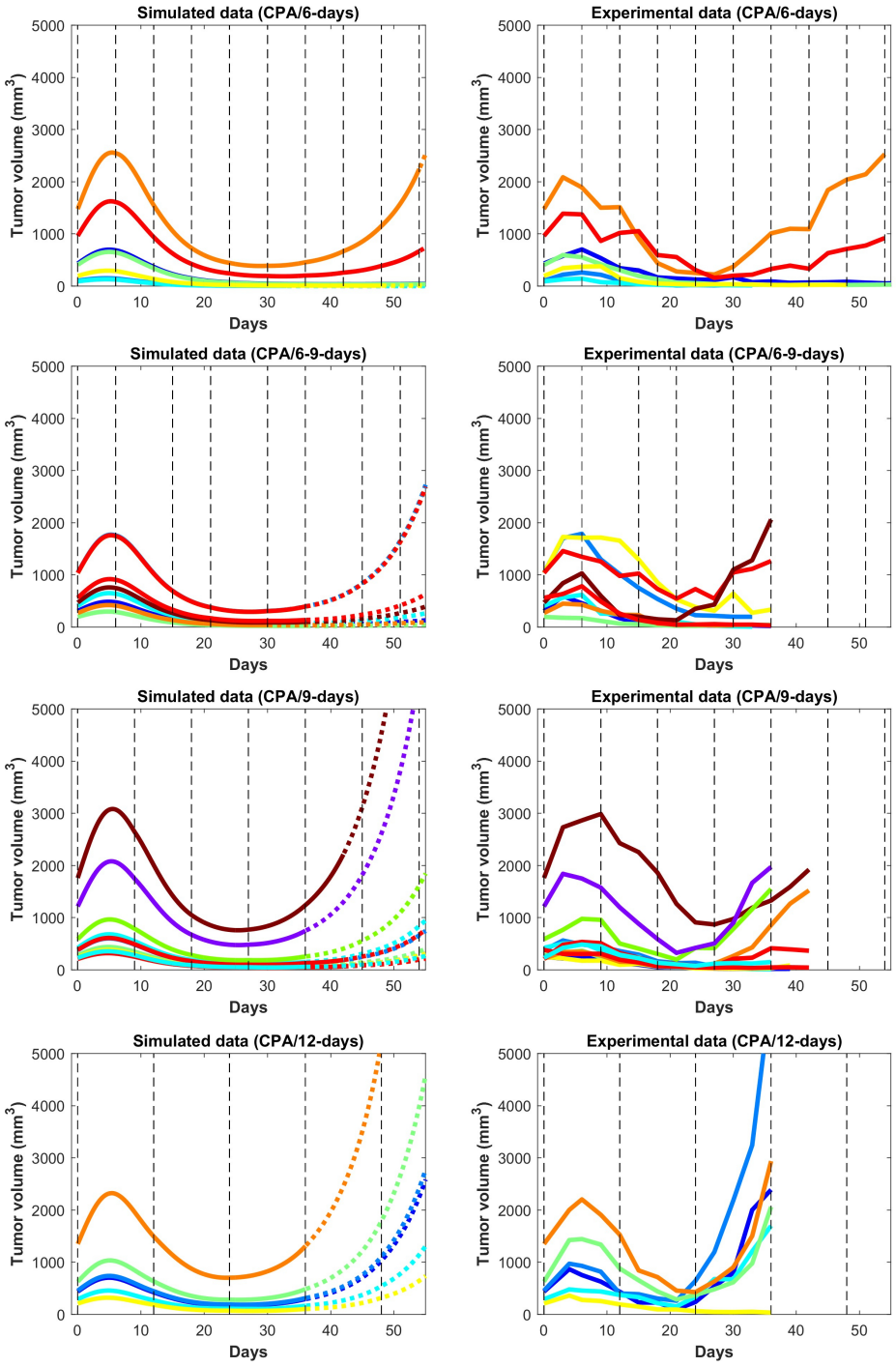


Figure 3.5: Simulated and experimental growth curves for the scenarios CPA/6-days, CPA/6-9-days, CPA/9-days, and CPA/12-days. Solid lines represent fitted data; dotted lines represent predicted growth curves extrapolated from the model.

Despite the use of pooled mouse data for curve fitting, there is a high degree of agreement between experiments and model fits, particularly for tumors that remained

largely suppressed throughout treatment. The nature of rebounding (or escaping) tumors makes the observations very stochastic in nature, while the models are progressively transitioning between regressing and rebounding tumors under conditions as the initial tumor volume gets larger (an example of this transition being the CPA/6-9-days treatment). Allowing for small variations in the parameters between individual experiments is likely to account for such variability. The primary intent is to showcase that one unique set of population parameters can capture the qualitative and quantitative behaviors of a large set of different metronomic chemotherapy treatments that involves induction of anti-tumor immune responses and drug resistance.

The agreement between simulated and fitted experimental growth curves is high for the untreated, 2-CPA, 3-CPA and CPA-9days(210mg/kg) cases, as can be seen in Fig. 3.4. There is a discrepancy between the model prediction and the experimental data for the largest tumor in the 2-CPA scenario. In the repeated treatments, the model captures well the progressive apparition of rebounding tumors as intervals between drug administration increase from 6 to 12 days in 4 increments. For the CPA/12-days scenario, the model predicts that tumor escape will occur up to ten days after when it actually occurred experimentally; nevertheless, the model is able to capture the qualitative effect of this treatment regimen, which fails rapidly within the first 3 drug injections.

The 1-CPA set of data was small in size ($n = 5$, before an outlier was excluded) and tumors that were implanted apparently grew at a significantly faster rate than the rest of the dataset. Given that each metronomic scenario was from a different batch of mice, systemic experimental variations in the fitted data can account for some of these observed discrepancies and can be hard to distinguish from model deficiencies. From Fig. 3.6, the three treatment conditions of 1-CPA, CPA/9-days and CPA/12-days are not expected to differ until day 9, but the plots indicate that the experimental data is inconsistent. In addition, 1-CPA and CPA/12-days have mice undergoing the same treatment conditions until day 12.

In Fig. 3.7, the mean normalized tumor volumes are given for each of the conditions modeled and shown with the corresponding experimental data. There is a slight

delay in the 2-CPA case when it comes to the rebounding behavior of the tumors. The CPA/12-days discrepancy in the time of rebound can also be seen. Notably, the deficiencies in the fits are more pronounced on the marginal cases of longer breaks and shorter treatments. It is possible that these edge cases require special attention to incorporate the impact of other phenomena that are discussed in [2], and might require additional experimental data to appropriately capture the underlying mechanisms. Also shown in Fig. 3.7 is the difference between the normalized average simulations of the fitted data without outliers and the experimental data with outliers. The trends were not affected by removing outliers with the exception of CPA/12-days that yielded large discrepancies between 4 and 25 days. This was due to two small tumors that were excluded as they grew very rapidly to be much larger than other tumors that had higher initial volumes.

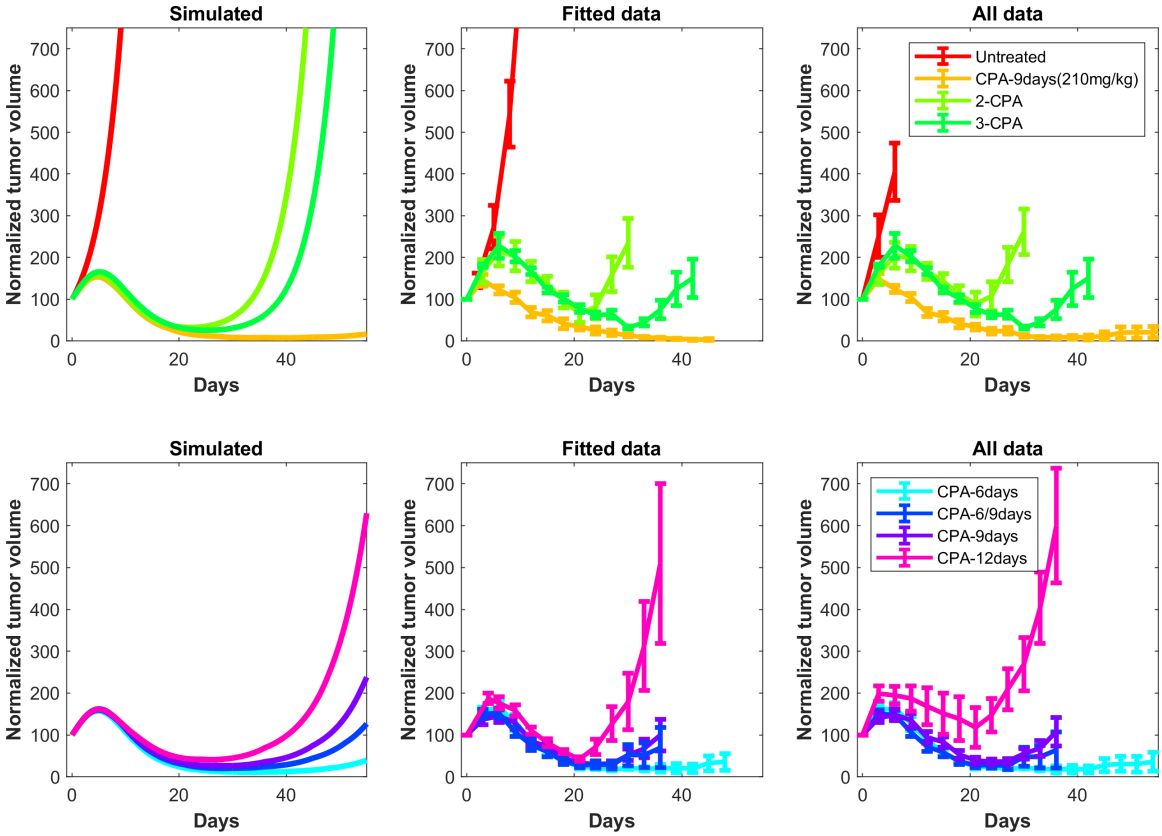


Figure 3.7: Simulated tumor growth curves of all the considered treatments scenarios were normalized such that the first time point of each curve has an initial value of 100. The mean of the normalized curves for each treatment condition was then calculated and plotted on the left panels. The right panels are the original experimental data from [3]. These data also show the difference between the normalized average fits of the fitted data without outliers (middle panels) and the experimental data including outliers (right panels). The outliers are shown in Fig. 3.3.

3.3.2 Predictions regarding the immune system

In Fig. 3.13, predictions are made for 1-CPA, 2-CPA, 3-CPA for the immune system behavior. Notably, the immune data were never used in model fitting, but model predictions of the immune system correspond well to that of the immune cell data reported in [3]. In this model, the immune system is assumed to be an aggregate of multiple immune cells and related factors. The experimental immune data, shown in Fig. 3.13, can be separated into two categories: the immune cell markers and the chemokine, cytokine, and adhesion molecule markers. It is clear from the experimental

data that new injections of the drug have a short term negative effect on immune cell populations. However, this is not often the case for cytokines or chemokines, which can be recruited and remain at high levels even after a second injection, as was the case for the 2-CPA regimen. Considering both effects of the immune cell markers and cytokines and related molecules, the predictions of immune system behavior by the model seem to be an aggregate of these responses. It estimates well the immune cell peaks that occur at 12, 18 and 24 days for 1-CPA, 2-CPA, and 3-CPA regimens, respectively. Notably, the experimental data are quite sparse, as only 4 time-points were analyzed for each treatment condition. Additional plots in Fig. 3.8 show the predicted immune system behavior for other treatment conditions.

3.3.3 Predictions for CPA/9-6-days and CPA/7.5-days

The validated model was then used to make predictions about the effect on tumor growth of dose administration regimens that were not experimentally tested. One such example is shown in Fig. 3.14, where drug is administered at alternating breaks of 9 days and 6 days (CPA/9-6days). Interestingly, the model predicts that CPA/9-6days is considerably inferior to the CPA/6-9days regimen, suggesting that shorter breaks between drug administrations early on improve outcomes, as compared to longer breaks. CPA/7.5-days is a little better than CPA/9-6days, especially with smaller initial tumor volumes.

We hypothesize that shorter breaks early on allow sufficient tumor burden reduction to enable cytotoxic immunity to have greater impact on the smaller tumor. Notably, within this framework, the standard approach of maximizing tumor burden reduction would cause excessive damage to the immune system, so chemotherapy-induced tumor burden reduction should be sufficient to augment the effect of the immune system but not to act to its detriment. Based on our analysis, although the CPA/9-6days schedule (Fig. 3.14) should be superior to the CPA/9days schedule (Fig. 3.5), the two simulated datasets are almost identical in behavior.

Upon closer examination of relative impact in cancer cell death due to activity of the immune system vs due to the drug, the model indicates that even after the

anti-tumor immune recruitment decreases, small tumors remain under control as a result of drug cytotoxicity. However, for large tumors, even a slight decrease in anti-tumor immunity can determine the difference between tumor regression and tumor progression. Looking at the immune response in Fig. 3.14, a slight decrease around 15 to 20 days after the first treatment leads to strong rebounding behaviors in the two largest simulated tumors.

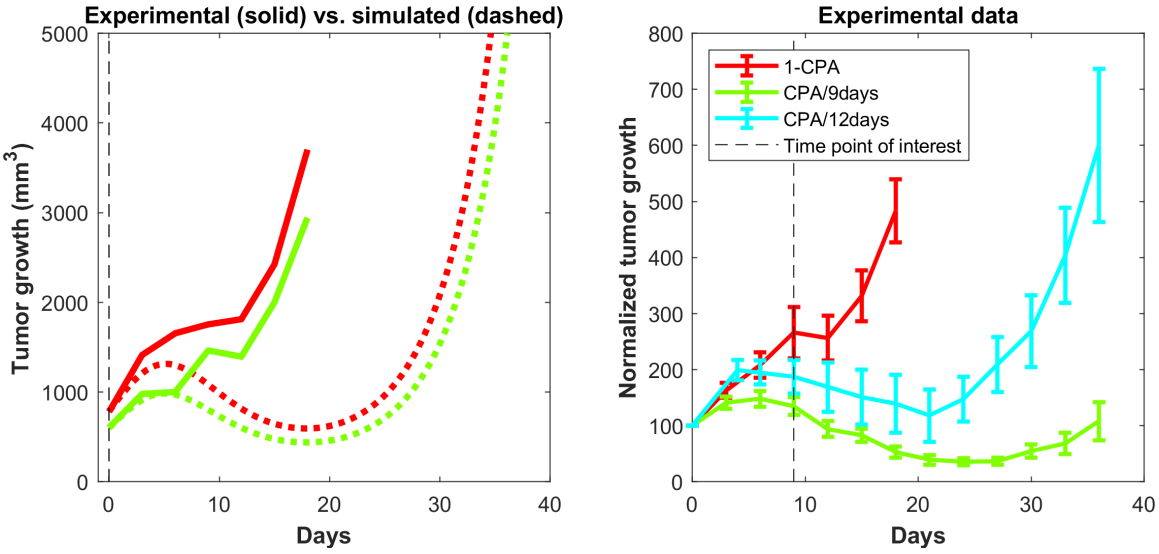


Figure 3.6: Discrepancies in modeling 1-CPA. On the left, are shown the experimental data and the simulated data, with the latter data predicting much slower tumor growth than was seen in the experimental data. On the right, the 1-CPA data is plotted using the normalized tumor growth and compared with CPA/9-days and CPA/12-days treatments. Note that the 1-CPA treatment is equivalent to the CPA/9-days and CPA/12-days data up to treatment day 9 and treatment day 12, respectively, barring systemic errors in the experiments.

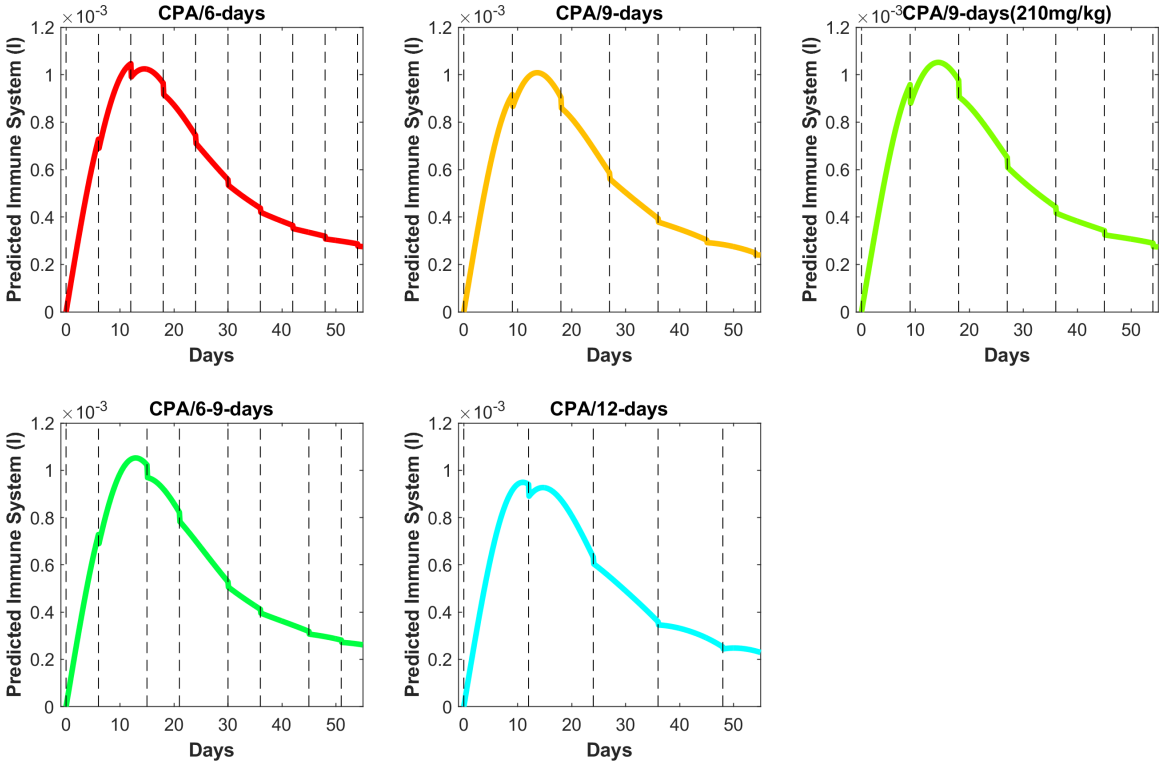


Figure 3.8: Predicted immune system from the model fits for the treatment conditions other than 1-CPA, 2-CPA, and 3-CPA.

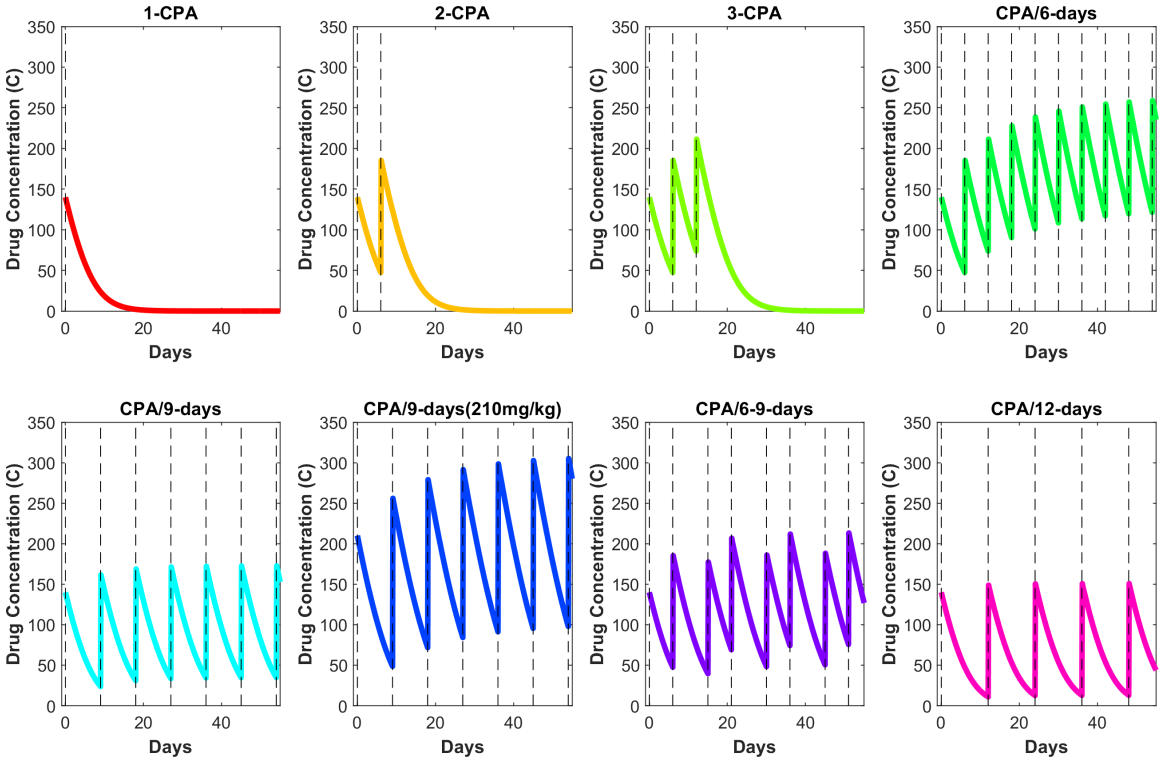


Figure 3.9: Prediction of the drug concentration (C) for all the experimental conditions in [3]

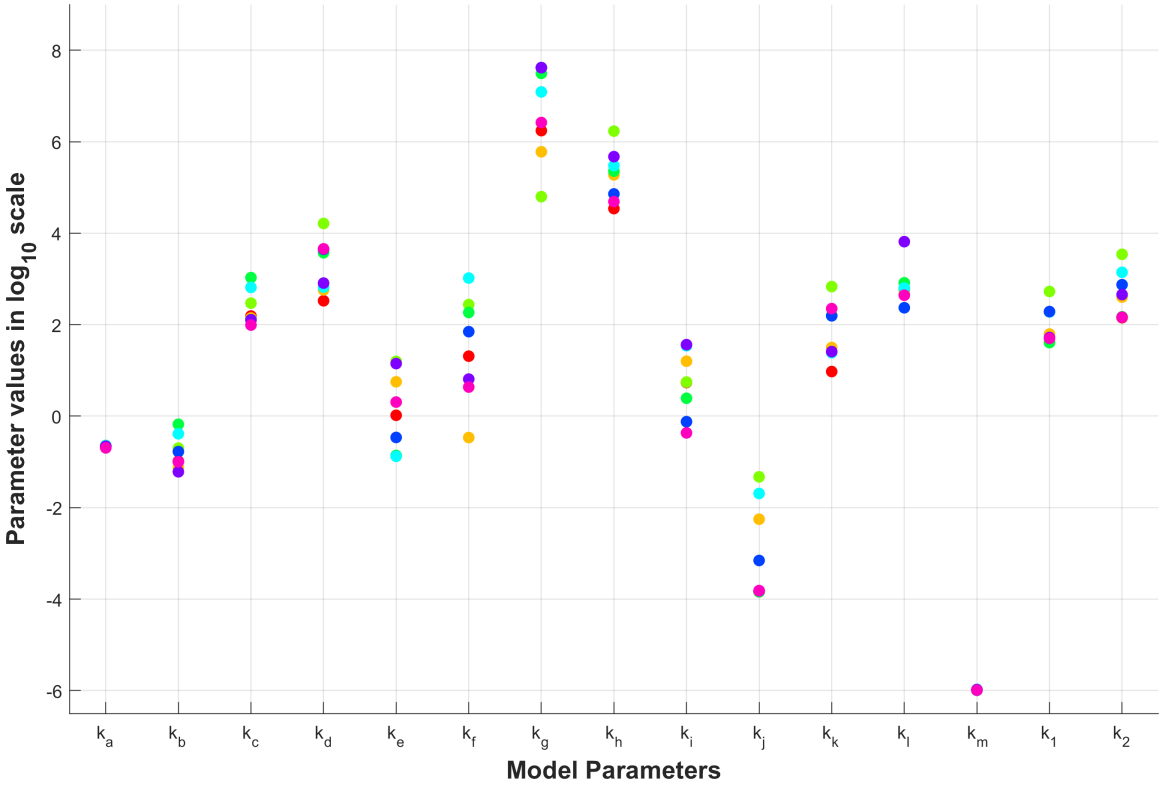


Figure 3.10: Parameter values in a log_{10} scale for 8 different fits within 1% of optimal value found for the objective function. Missing dots are due to overlapping parameter values, which can be found in Table 3.1

3.3.4 Sensitivity Analysis

The sensitivities are computed using the package *safe*[4], using specifically the Fourier method described in [112]. For each parameter, and for a suitable hypercube in parameter space centered at the nominal parameter set (in this case, we allowed a 10% variation on each parameter) this package reports a number between 0 and 1, at each time point, that measures the contribution of that parameter to a change in the model output. The sensitivity of the model output to parameter i is then calculated as the partial variance, defined as the ratio σ_i^2/σ^2, where σ_i^2 denotes the power in the spectrum of the i-th parameter, and σ^2 the total power.

These calculated sensitivity indices are shown in Fig. 3.11 for Fit A. Across the 6 different condition shown, the parameters k_c, k_e, k_f, k_j, k_l, and k_m are shown to be less sensitive or completely insensitive to variations of 10% in these parameter values. In further investigations of this model, these sensitivity indices suggest possible improvements by adjusting the range of parameter values explored or simplifying of the model by removing some of these terms if they do not contribute significantly to the model accuracy. This could also suggest that some of these terms of the ODE model containing these insensitive parameters need to be adjusted to better capture the overall dynamics of the tumor volumes. Looking at a similar analysis for Fit G in. 3.12 which had parameter values most different from Fit A, the sensitivity indices are shown to be relatively similar with the exceptions of the parameters k_c and k_g. k_c is more sensitive in Fit G, while k_g is more sensitive in Fit A. Thus, it may suggest that there may need to change the functional forms of the two terms containing these parameters. For both fits, the sensitivity indices were consistent across all the different treatment conditions.

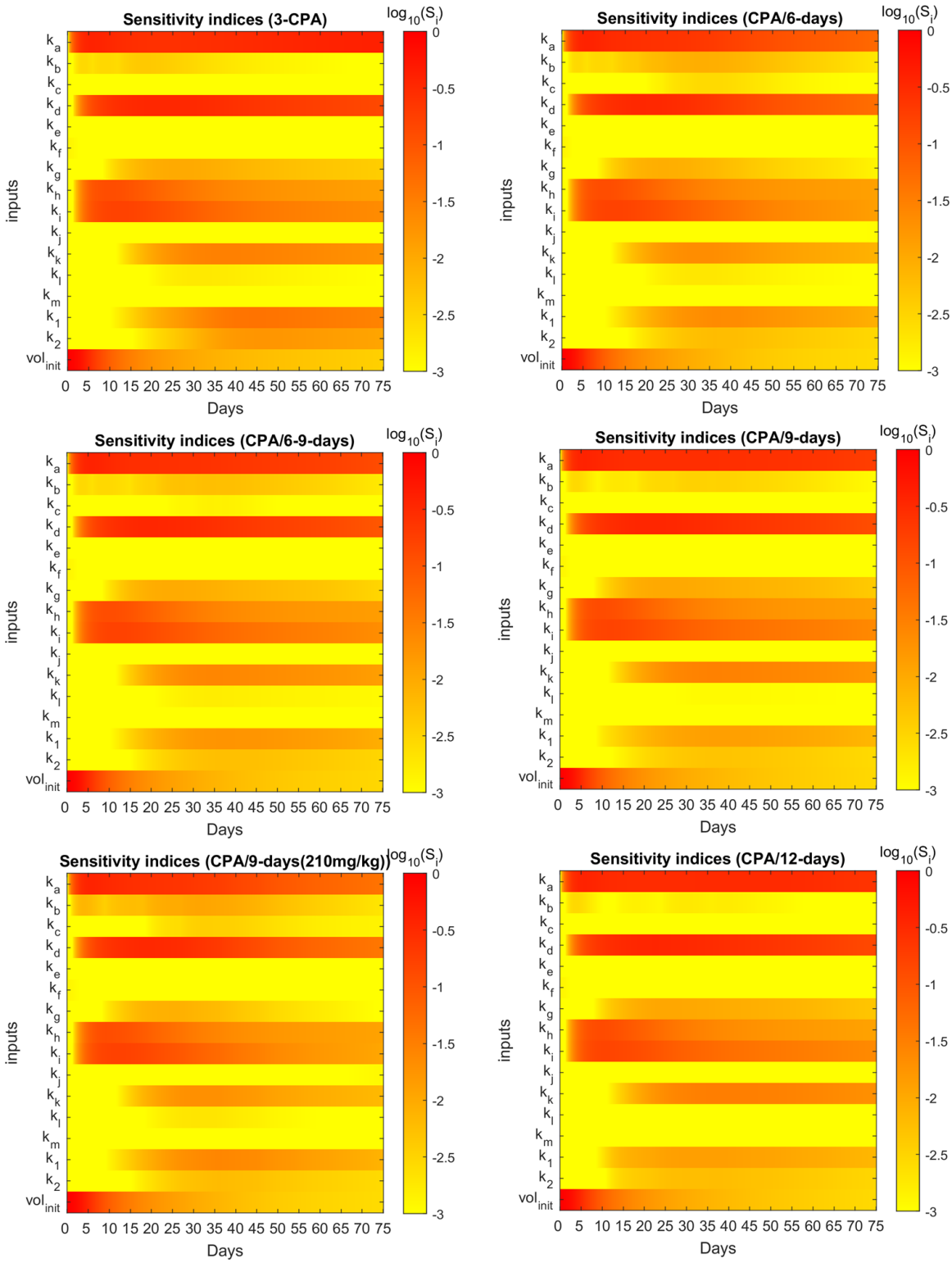


Figure 3.11: Time-varying sensitivity indices on a log_{10} scale using the FAST method described in [4] for Fit A.

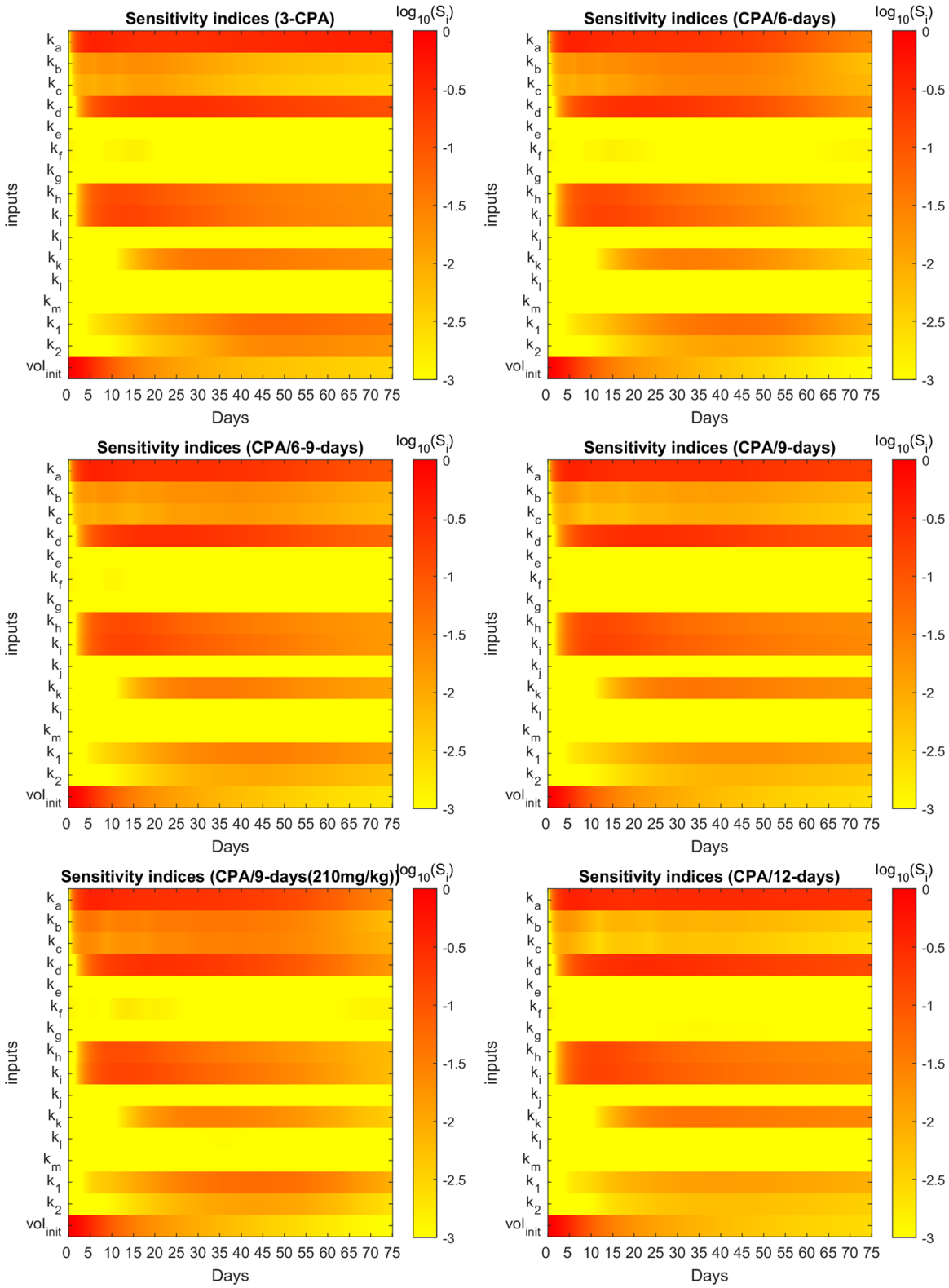


Figure 3.12: Time-varying sensitivity indices on a log_{10} scale using the FAST method described in [4] for Fit G.

3.4 Discussion

The administration of cyclophosphamide under MTD regimens can be severely toxic to the immune system and undermine its ability to help control tumor growth. Changing the dose and timing of drug administration has been shown to restore the ability of cytotoxic immune cells to target tumor growth in glioma mouse models [3, 15], suggesting the existence of a "sweet spot" that can minimize damage to the immune system and maximize anti-tumor immune effects. To formalize the mechanisms that may underlie experimentally observed variation in response with respect to drug dose and schedule, we propose a phenomenological mathematical model that captures key processes that may underlie interactions between drug, immune system, and tumor. Through these efforts, we aim to identify key mechanisms that may give rise to a strong and sustained immune response, within the broader context of improving the understanding of cancer treatments that target not only cancer cells themselves but also the tumor microenvironment, and in particular, the immune system.

The proposed phenomenological model was fit using a single set of parameters that was able to capture well tumor growth dynamics across nearly all experimentally tested drug treatments (Figs. 3.4 and 3.5). The model was further validated through predicting the dynamics of the immune cells that were not used in the fitting process. We found a strong correspondence between immune data and predictions, despite the simplified nature of the underlying model (Fig. 3.13). The parameter values of various fits within 1% of the optimal value for the objective function are plotted in Fig. 3.10. Notably, the tumor growth constant k_a was consistent among all the fits. This parameter is likely to be well-determined due to the abundance of tumor volume data and, in particular, the use of untreated tumor growth data in the fits. The parameters related to the tumor equation and the PK part of the model are generally within one order of magnitude among all the fits. The remaining parameters are mostly spanning several orders of magnitude and it is likely that a better correspondence of phenomenological variables X, Y, and I with the experimental data would alleviate this issue, namely that several combinations of parameters can

redundantly capture the qualitative behaviors seen in the experimental data.

We then used the validated model to predict the impact on tumor growth of an alternative schedule of CPA/9-6days as well as CPA/7.5-days that have not been tested experimentally (Fig. 3.14). The model predicts that shorter breaks between dose administrations early on lead to greater tumor burden reduction and improved anti-tumor immunity; however, as was shown in [3, 15], the breaks cannot be too short, which in turn may lead to excessive immune cell depletion, not allowing cytotoxic lymphocytes time for replenishment. Therefore, the goal of "sweet spot" therapy is to reduce tumor burden sufficiently to allow cytotoxic immunity to persist and control the tumor.

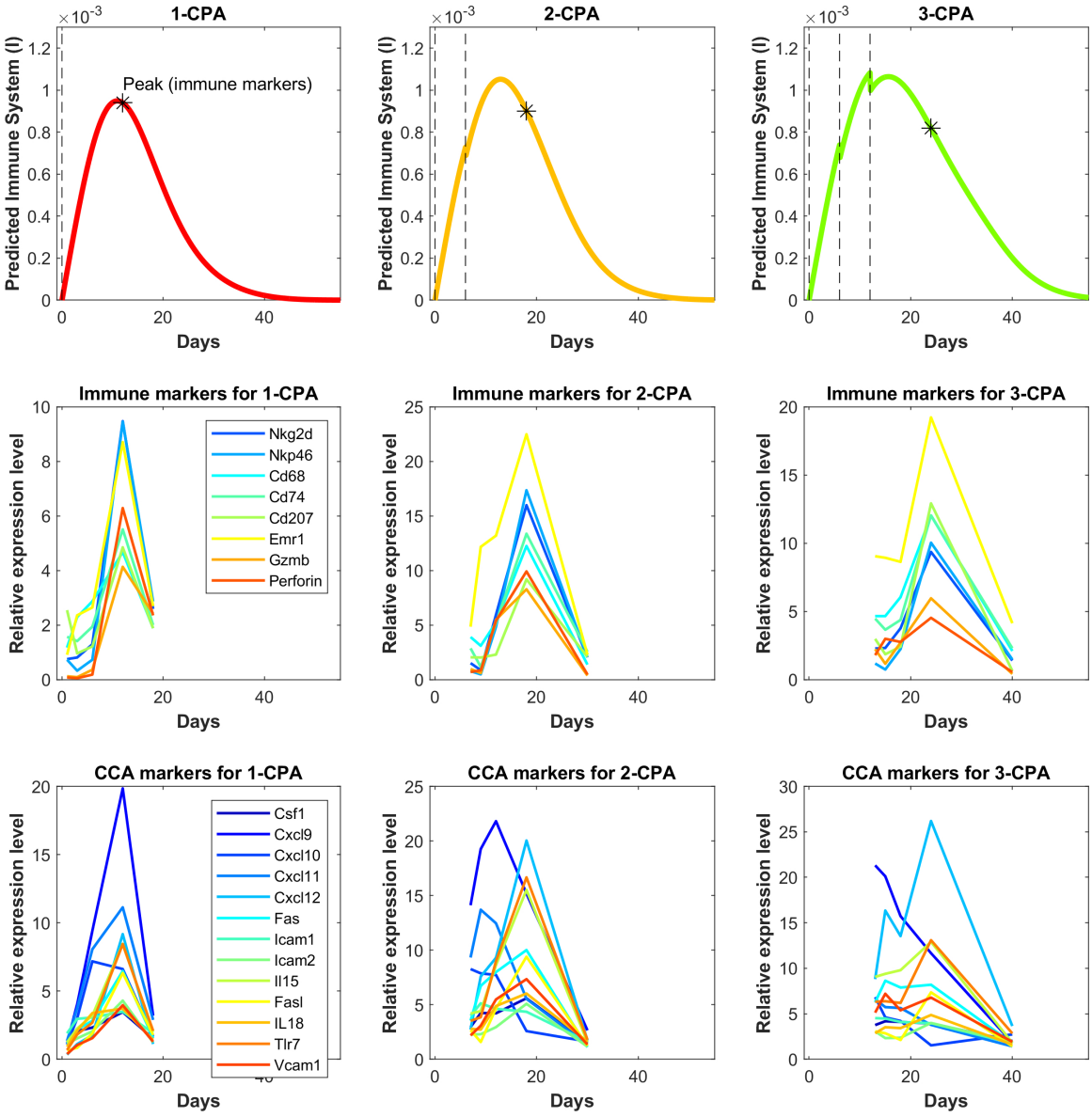


Figure 3.13: Independent model validation through comparison of model predictions to available immune cell data. The top row shows the predicted immune system recruitment by the fitted model for the 1-CPA, 2-CPA, and 3-CPA treatment conditions. The vertical black dashed lines indicate times of drug injections at 140 mg/kg. The initial volume for the simulated tumors yielding these curves was assumed to be 1000 mm^3 at the time of the first injection. The middle row shows the experimental data of the gene expressions for various immune markers linked to the innate immune cells (macrophages, dendritic, and NK cells). Similarly, the bottom row shows gene expression data for various chemokines, cytokines, and adhesion molecules (abbreviated CCA on the plots). The data is ordered such that each column represents the same treatment condition.

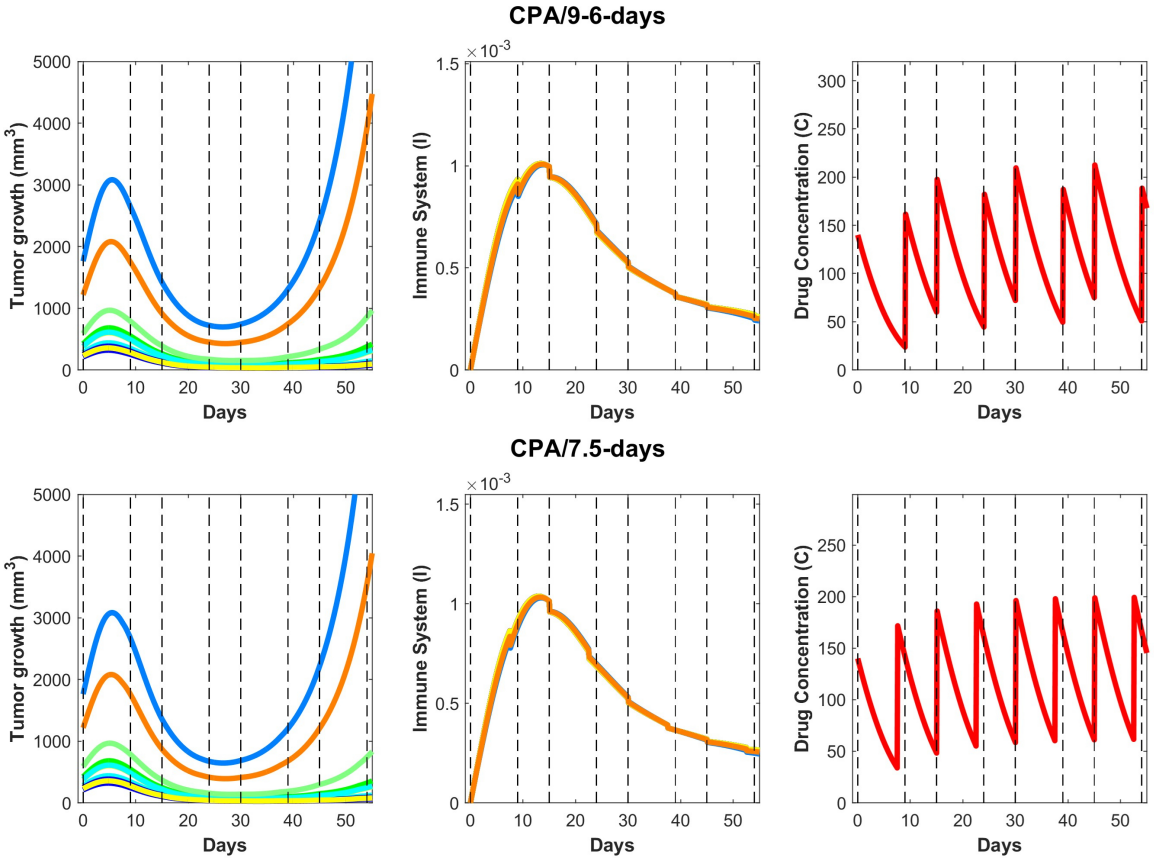


Figure 3.14: On the top row, predictions for the CPA/9-6-days drug regimen where the first break is 9 days followed by 6 days, then 9, then 6 and so on, alternatively. On the bottom row, similar predictions are made for CPA/7.5-days.

Despite the undeniable complexity of the immune system, the proposed conceptual model of four differential equations coupled with a 1-dimensional PK model allowed us to capture tumor responses to various treatment schedules. While the model was used to understand and reproduce data that shows impact of variation in dosing and scheduling on tumor growth for a particular mouse tumor model, it also highlights the fact that beyond simply understanding the interactions between the different components at play, a quantitative modeling approach may be able to help design better metronomic chemotherapy treatments. Furthermore, the conceptual nature of the proposed model enables us to pinpoint more specific mechanisms that are responsible for observed variations, which may not be possible with more detailed descriptive models.

Besides fitting well the experimental data on cyclophosphamide, the mathematical

model developed here can be a valuable complementary tool to understand how drugs function and how they can be combined with other types of treatments, such as immunotherapies. It is also possible that drugs previously discarded due to a lack of cytotoxicity may have immunogenic properties that could act as effective complements to other treatments, a response that may have gone unnoticed in the context of MTD administration. Furthermore, the proposed framework suggests that what may appear as therapeutic resistance to a drug may in fact represent desensitization, a phenomenon that can be mitigated through alterations of the drug dosage and scheduling and through better understanding of how these different components of the tumor microenvironment interact with one another, particularly immune cells and cytokines.

One of the exciting properties of this model is that it demonstrates that the exhaustive details of specific immune cell and cytokine types are not necessary to explain experimentally observed data with regards to tumor growth; instead, we showed that grouping actors by function into "classes of cells" is surprisingly sufficient. Therefore, while it is tempting to try to introduce more details into equations describing either the immune cells or immunostimulatory and immunosuppressive factors, it may not provide additional insights. This is because the ultimate goal of this approach is not to quantify the immune system extensively, but to reliably predict long term tumor dynamics under treatment. Instead, it may be more beneficial to explore the impact of drug-specific kinetics on these different components. This approach is quite feasible, since pharmacokinetic modeling that describes experimentally measurable changes in drug concentrations over time has very well-developed methodology that is used extensively in drug discovery and development.

Drug-specific PK models can be coupled with the four Eqs. (2-5) to help better understand the push and pull of drug effects on all system components to get the ultimate measurable outcome: drug impact on tumor growth. Parameters that would need to be experimentally determined are those associated with drug-dependent toxicity on immune cells and cancer cells separately (within the frameworks of this model, they are parameters k_b, k_f, k_m, k_i and k_l), which can be measured in targeted ex-

periments and then used as starting points for parameter optimization. The PK-PD model will then need to be trained on experimental data, as in the approach described here, and then used to evaluate the impact of different treatment regimens on final tumor growth. Notably, PK models can be created to describe the dynamics of both novel drugs and existing ones. This may open avenues to re-purposing existing drugs by appropriately altering the dosage and schedule of administration, an undertaking where quantitative approaches such as the one proposed here may prove to be indispensable.

The effectiveness of metronomic CPA schedules examined here has been demonstrated in syngeneic mouse glioma models, as well as in innate immune-sufficient scid mice in work that includes mouse, rat and human xenografts; however, we do not know how predictive these models are of responsiveness in gliomas or in other tumor types in human patients. Evaluation of these metronomic CPA schedules in human tumor models, including patient-derived xenografts, is therefore a high priority for clinical translation.

We anticipate the proposed mathematical model will be useful for discovery of hidden potential of current drug treatments by building better quantitative understanding of the phenomena at play, and for designing more effective drug regimens that may not be intuitively apparent through exploring immunogenic potential of ICD drugs. The proposed model can also be used to investigate a variety of drug schedules and dosing regimens, as well serve as a building block for investigation of combination chemotherapy and immunotherapy treatments that will likely pave the future of cancer therapy.

Part II

Decision making by immune cell populations through distributed computation

Chapter 4

Introduction and Literature Review

4.1 Distributed cellular computation in immunology

Immune cells can inherently perform intricate operations to defend the host against threats. This is enabled, in part, by the fact that immune cells can communicate with one another in a circuit-like manner via a diverse set of communication mechanisms such as cytokines and chemokines, an example of which is shown in Fig. 4.1. Hence, it has been long recognized that biological circuits resemble electronic circuits in their ability to process logical operations [114]. The implementation of synthetic circuits for biological computations inside immune cells [115] is of particular interest in cancer

therapies such as immunotherapy [116, 117]. However, the engineering of anti-tumor immune cells remains at an early stage and the current technology remains mostly limited to the implementation of logic circuits within one type of cell at a time. Before one can think of engineering a more complex network of logic interactions, it first needs to work in individual cell types. The most promising results have been observed in the area of adoptive T cell therapy.

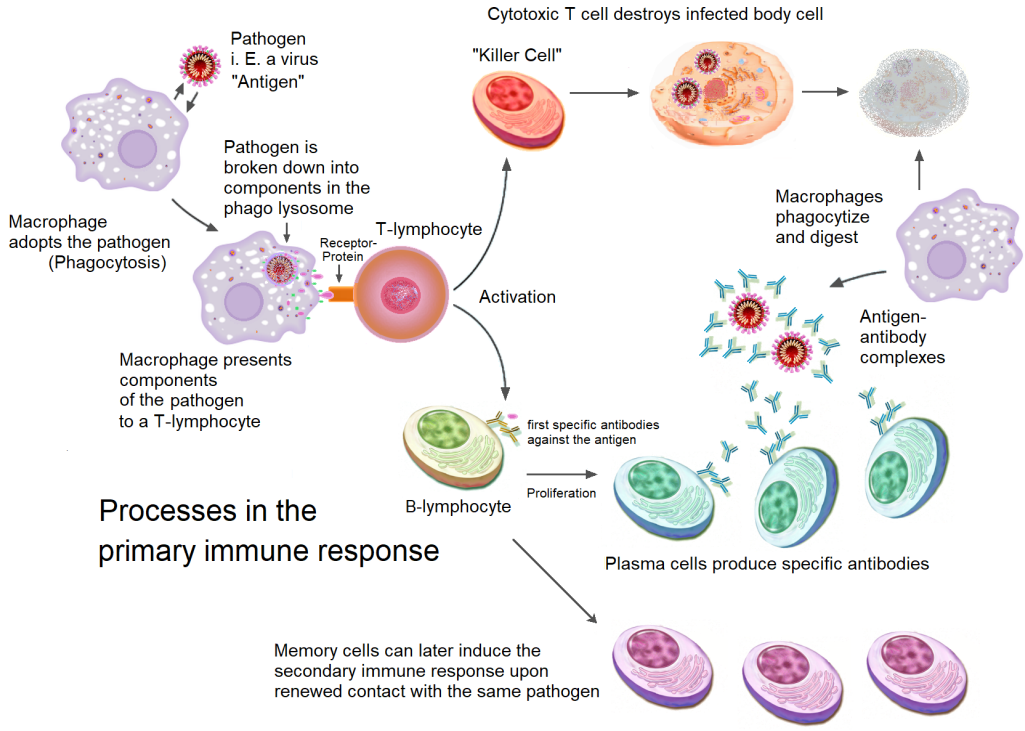


Figure 4.1: Schematic of the processes in the primary immune response [5].

In the continuity of studying the intricate relationship between cancer cells and immune system, this branch of immunotherapy leverages the ability of engineered cells to carry out complex computations and responses to properly screen their desired targets. In adoptive T cell therapy, T cells are taken from a patient and the specificity of these T cells is modified to incorporate the genes that encode for the receptors that target the cancer cells found in the patient (shown in Fig. 4.2). These modified T cells are then transfused back. These therapies come in two engineered forms: T

cell receptor (TCR) or chimeric antigen receptor (CAR) [29]. CARs are engineered receptors that are designed to target a specific antigen on the surface of cancerous cells. TCRs, on the other hand, target epitopes. Both types of treatments have shown toxicity as some of the targets may also appear on non-cancerous cells and may, for example, attack vital organs. Adverse effects with CAR-T cell therapies have also been observed where the T cell response was found to be too strong and life-threatening [118]. In some clinical trials with CARs treating leukemia, severe symptoms from cytokine release syndrome have been observed [119]. Yet, the positive results have been so promising for the fight against cancer that there remains a large degree of enthusiasm about this approach. The main priority of the field has been to enhance the specificity of these engineered T cells to avoid targeting healthy cells.

It has been shown that using a combinatorial activation system of two antigens improved the specificity towards targeted cancer cells because bindings to both receptors are necessary to drive activation and proliferation of the T cells (an "AND logic"). In a GoCAR-T treatment, a CAR domain is coupled with a costimulatory domain that binds to rimiducid thus allowing control over the activation process [120]. In yet another case, cancer-specific antibodies are attached to a CD16 domain that triggers T cell activation (antibody-coupled T cell receptor or ACTR) expanding greatly the available library of antigens that these T cells can bind to [121]. There is also the possibility to add chemokine receptors onto the T cells to guide them towards the tumor site through chemotaxis or using tumor-derived signals [122]. There is interest in developing kill switches to stop the therapy if the health of the patient is at risk and in modulating the immune response in terms of when it is active and limiting its magnitude through negative feedback mechanisms [123, 124, 125, 126, 127]. These improvements are briefly depicted in Fig. 4.3. One can also consider controlling the growth of T cells through a drug-inducible mechanism. There is also the importance of activating different costimulatory domains leading to different T cell responses or whether the engineered T cells are mostly in their naïve or activated state is also of importance depending on the application. These added complexities to the traditional CAR treatment have, for the most part, been studied one at a time but there

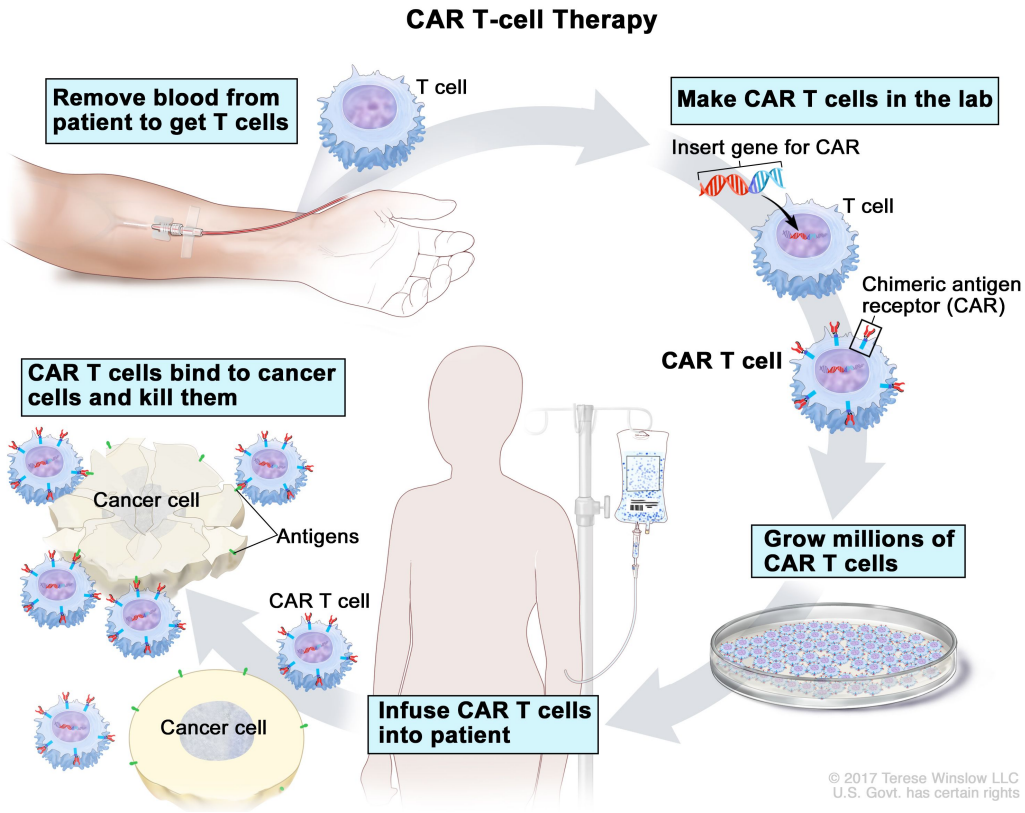


Figure 4.2: A schematic of CAR T-cell therapy [6].

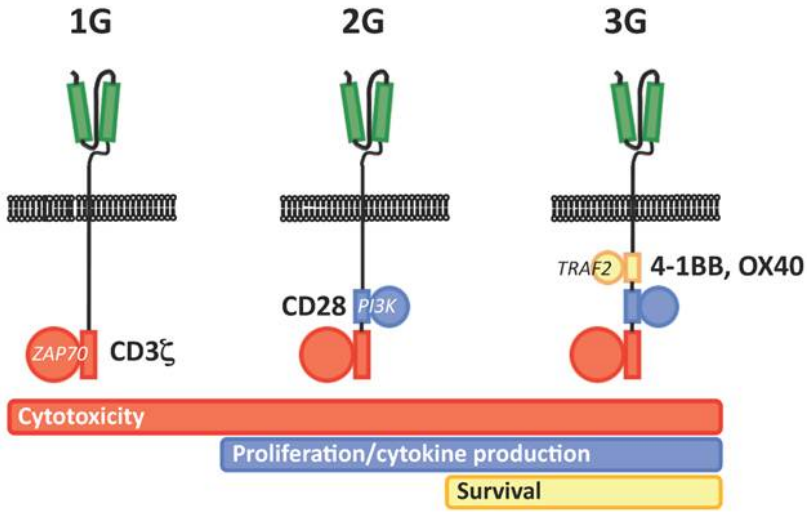


Figure 4.3: Depiction of first, second, and third generation CARs [7].

are efforts to combine these ideas such as the split, universal, and programmable (SUPRA) CAR system allowing various modular behaviors like an "OFF switch", an "OR logic" in choosing targets, or selecting target cells based on the number of available antigens [128]. CAR-T cells are not the only form of engineering immune cell therapies with appearing therapies using CAAR-T (with a chimeric receptor with an antigen-binding domain targeting autoreactive B cells), CAR-Treg cells, and CAR-NK cells [129]. This sophistication and expanding array of tools in immunotherapy emphasizes the importance of designing logic circuits that can utilize these various functionalities.

4.2 Current biological circuit design and distributed computation

Despite its potential, the rational design of biological circuits was initially very difficult and labor intensive, as it required striking a precise balance of regulator abundances. In addition, the sensitivity of cells to growth conditions was poorly-understood, and it was difficult to characterize cell behavior [130]. Furthermore, in the context of bacteria, the expansion of the set of transcription factors (TFs) with negligible cross-talk that perform reliably under a wide range of conditions has been challenging [131, 132, 133]. These challenges are even more difficult in mammalian cells [32].

Recent biotechnological developments have ameliorated some of these difficulties. Firstly, computer-aided techniques have been developed to automate the design process [30]. Secondly, efficient and reliable TFs can now be designed through re-purposing of the CRISPR associated protein, known as Cas9, by deactivating its endonuclease activity [134]. This catalytically-inactive Cas9 known as dCas9 retains the ability to bind to a single guide RNA (sgRNA). The resulting complex acts as a TF by binding to the region on the genome that matches its sgRNA strand. The bound sgRNA-dCAS9 complex turns the targeted gene off by repressing the adjacent

promoter or by interfering with the elongation downstream [135]. This new platform, referred to as CRISPR interference (CRISPRi), has a modular nature that enables a single protein dCas9 to target multiple genes at once by changing the associated sgRNA [136]. CRISPRi has been used to build multi-input genetic circuits that consist of NOT and NOR gates [136, 32].

The CRISPRi technology allows the design of an extremely large number of orthogonal regulators based on sgRNAs. A limiting factor is the fact that dCas9 is a shared resource amongst the different gates which needs to be continuously expressed at very high concentrations, and this leads to high toxicity for the host cells [136]. This problem is known collectively as the *dCas9 bottleneck* and has been the subject of a high level of research activity aimed at reducing dCas9 toxicity [137, 138, 139, 140, 141]. Such practical limitations have severely limited the number of repressor-based gates that can be implemented together in a cell. Hence, the maximum reported number of repressor-based gates per cell has not exceeded 7-8 gates in both *Escherichia coli* [30, 31] and yeast [32]. The typical numbers are even lower than that. For instance, some of the repressor-based circuits published in the last two years reported circuits with four gates [142], fives gates [143], and six gates [144] per cell.

The difficulty in designing scalable biological computational systems has prompted the development of distributed approaches in synthetic biology [145, 146]. Bacteria can exercise cell-to-cell communication via diffusible small molecules (DSMs) [147], [148] or conjugation [149]. In our context, the problem of TF bottlenecks can be circumvented by distributing the overall computation among several (colonies of) cells, each of which performs a specific part of the computation, thus allowing the same TFs to be reused, and keeping dCas9 concentrations at innocuous levels. This approach has been demonstrated by the construction of all types of two-input Boolean functions in cells, each containing one gate [150]. A rule-based approach to distributed design of simple Boolean functions via AND/OR/NOT gates has been proposed [151]. More recent designs has cells, each implementing a 1-input-1-output IDENTITY/NOT function, used as building blocks in segregated consortia [152]. A

similar approach uses IDENTITY/NOT building blocks in a hierarchical distributed scheme via recombinase logic without utilizing cell-to-cell communications and without formally studying the limits on the number of gates per cell [153]. Other recent designs employing DSMs include a 3D cell consortium which functions as a full adder [154], a multiplexer-demultiplexer system [142], coupled incoherent feedforward loops [155], and cascades of toggle switches [156].

In this work, we are interested in developing a synthesis framework for systems using DSMs. A minimal unit consists of two components: a "sender" consisting of the enzyme that produces a DSM and a "receiver" that responds to this DSM and turns on a promoter. However, the number of sender/receiver pairs that have been used together remains very small. Similar to crosstalk between TFs, there is the additional challenge of building an orthogonal set of DSMs where each receiver only responds to its cognate signal. Using directed evolution approaches [157], the method described in [158] can be applied to create a set of four orthogonal sender/receiver pairs (C4-HSL Rhl, 3OC12-HSL Las, 3OC6-HSL Lux and pCou-HSL Rpa). Efforts to expand the set of the orthogonal DSMs to eight are underway.

The aim of this work is to automate the process of Boolean synthesis for diffusible systems, under constraints on the number of available orthogonal DSM molecules and constraints on the number of gates per cell. In state of the art technology, the feasible numbers are around seven gates per cell and a total number of 4-5 DSMs.

4.3 A novel Boolean synthesis problem

Due the nature of the CRISPRi framework and limitations of the current technology, we assume that the Boolean synthesis problem is constrained to use 2-input NOR (NOR2) gates only. (Note that a NOT gate can be implemented as a NOR2 gate with identical inputs.) NOR gates are universal, in the sense that any Boolean function can be represented via NOR gates [159]. In order for cells to communicate with each other, DSMs are used in our design. A cell can release one or multiple DSMs that can act as inputs to other cells or as overall outputs of a circuit.

Given a Boolean Function (BF) and its full circuit representation, the most natural "top-down" approach is to apply graph partitioning algorithms that, in essence, cut the full design into smaller subdivisions [160, 161]. However, these partitioning methods tend to yield designs that require a large number of cuts in order to fulfill the constraints on the number of allowed gates per cell (and thus, requiring the implementation of a large number of DSMs).

Furthermore, since the small molecules are diffusible, if multiple cells release the same DSM, the released concentrations add up, and the DSM will in effect act to implement an "virtual OR gate". If a cell has a DSM input, then the logical value of that input is the sum of all the logical values of the outputs of all other cells that release that DSM. These particularities call for an alternative approach.

Our work uses elements of circuit design theory, combined with the adaptation of a branch-and-bound algorithm originally developed in a different context by one of the authors [1] to build a database of optimal NOR2 representations for all Boolean functions of up to 4 inputs. In comparison, the Cello software [30] produces NOR2 realizations via popular packages such as ABC [162] and Espresso [163] which are not guaranteed to be generate circuits which are optimal (in the sense of having a minimal number of gates) [164].

We will formulate the problem next. Let n be the number of inputs, and p be the number of outputs, such as reporter signals. We are given two positive integers N, q, which are the maximum number of NOR2 gates per cell, and the maximum number of orthogonal DSMs, respectively. Mathematically, a vector-valued (i.e, multi-output) Boolean function (BF) $F : \{0,1\}^n \to \{0,1\}^p$ can be thought as a vector $(f_1, .., f_p)$ of scalar-valued (i.e., single-output) BFs $f_j : \{0,1\}^n \to \{0,1\}, j = 1, .., p$. Each BF can be succinctly represented via a hexadecimal code (see Methods). Given N and q, we are interested in the problem of finding a distributed circuit realization that obeys the two following constraints:

1. each cell is constrained to use a maximum of N NOR2 gates,

2. a total of no more than q DSMs are allowed.

If a realization satisfies these constraints, then $F = (f_1, .., f_p)$ is said to be (N, q)-*realizable*. A mathematically precise definition is provided in the Methods section. In the remaining of the paper, a BF refers to a scalar (single-output) BF, while a multi-output BF is referred to as a vector BF.

4.4 Logic Synthesis

The translation of a behavioral model into a circuit layout using automatic synthesis is a central computer-aided design (CAD) problem. The problem of logic synthesis is generally divided into two categories: two-level synthesis and multi-level synthesis. Two-level synthesis is considered a mature field as the optimal implementation of a mapping with a large number of inputs can be computed in a reasonable amount of time [165, 166]. Early on, the Quine-McCluskey algorithm constituted an early effort at automation that was later replaced by the more efficient Espresso heuristic logic minimizer [167]. This is particularly useful when designing programmable logic arrays (PLAs) but typical circuit designs require multiple levels of logic. Due to the larger degrees of freedom that multilevel synthesis algorithms need to deal with, the search for an optimal solution is considerably more challenging, even for a 5-input Boolean function [168]. For these reasons, there has been little interest in recent decades in the area of optimal multi-level synthesis, with most of the contributions having been made in the 1950's through 1970's [169, 170, 171, 172, 173, 174, 175, 176, 177, 178, 179].

In contrast, the heuristic approach to multi-level synthesis has been a major focus of the field and makes a trade-off of the optimality of the solution in favor of other criteria and the ability to find large circuit implementations in a reasonable amount of time. In the 1970's with the IBM and *LSS* system [180], the rule-based approach was introduced and leveraged local transformations based on ad hoc rules and based on patterns. This had the downside of missing the global view of the synthesis problem. The 1980's saw the rise of the algorithmic transformations with algorithms such as *MIS* [181] and *BOLD* [182, 183, 184, 185, 165, 186, 187, 188]. Progressively, these algorithmic transformations were combined with the rule-based approach by using

the former in the initial stages of the circuit design and using the later in the later stages. Approaches based on this hybrid approaches include *SOCRATES* [182] and more recent version of *LSS* [187]. The *SIS* logic synthesis flow was built upon *MIS*, while bringing its own series of improvements [189, 190]. Around the same time as *SIS* in 1990's, the multi-valued logic synthesis system (*MVSIS*) [191] introduced the more efficient logic representations known as And-Inverter Graphs (AIGs) that later led to the developments of *ABC*, that took the best of *MVSIS* and added new features and improvements [35].

Chapter 5

Distributed implementation of Boolean functions by transcriptional synthetic circuits

5.1 Motivation of this work

In this work, after introducing a formal mathematical formulation of the problem, we propose an alternative "bottom-up" approach that builds up from the synthesis of individual cells using the idea of distributed computation. The next step is to take advantage of the additive properties DSMs in order to compute the output as a disjunction, thus limiting the number of DSMs required. We also propose a graph partitioning algorithm to complement the aforementioned method. Finally, we create a comprehensive optimized and automated design framework to select the

"best" design amongst the solutions found satisfying the mandatory bounds (number of gates per cell and number of total DSMs) . To that end, we propose several secondary criteria, to be chosen by the user, to get a unique solution. They include minimizing the number of gates, average number of gates per cell, or the range of the number of gates per cell.

5.2 Methods

5.2.1 Background on Boolean Algebra

A Boolean function (BF) f of n inputs is a mapping $f : \{0,1\}^n \to \{0,1\}$. Hence, there are 2^{2^n} BFs with n inputs. A BF can be represented by at least three equivalent formalisms:

1. By specifying its truth table, i.e., listing its values (zero or one) for each of the 2^n possible binary vectors of n inputs. This list can be written as a string of 2^n binary digits. Thus, when there are $n = 4$ inputs, we specify functions as a string of length 16. For example, the four-input function $f(x) = \bar{x}_1\bar{x}_2x_3\bar{x}_4$ is nonzero when the input vector is $(0,0,1,0)$, and is zero otherwise. Since $(0,0,1,0)$ is the third vector in the list of all 16 possible vectors of length 4, the function would be specified as $00\ldots00100$, with a '1' only in the 3rd entry from the end. A convenient shorthand is to use a string of $\lceil 2^{n-2} \rceil$ hexadecimal digits (so, four hex digits when $n = 4$). The example above would be described as 0x0003 (the "0x" prefix indicates the use of hex notations).

2. By writing a Boolean algebraic expression that gives the value for each choice of inputs.

3. By a providing circuit *realization* that shows the interconnections between the gates.

There are two dual standard Boolean algebraic representations of BFs: the disjunctive and the conjunctive forms. Here, the disjunctive form (DF) is considered

[159] which uses three Boolean operators: OR, AND, and NOT. For two Boolean variables x, y we use the following notation: $x + y := \text{OR}(x, y), xy := \text{AND}(x, y)$, and $\bar{x} := \text{NOT}(x)$. We also denote $1 := \text{TRUE}, 0 := \text{FALSE}$, $x^1 := x, x^0 := 1$, and $x^{-1} = \bar{x}$. With these notations, any DF can be written as follows:

$$f(x_1, .., x_n) = \sum_{j=1}^{m} \prod_{i=1}^{n} x_i^{\sigma_{ji}}, \tag{5.1}$$

where m is the number of terms in the disjunctive form and $\sigma_{ji} \in \{1, 0, -1\}$.

A *literal* is a Boolean variable that is uncomplemented or complemented, e.g x_i, $\bar{x}_i$ are literals. We say that a BF c is a *product term* if it is a product of literals. A product term is called a *minterm* of n variables if it has exactly n literals. Hence, a product term $\prod_{i=1}^{n} x_i^{\sigma_i}$ is a minterm if $\sigma_i \in \{\pm 1\}, i = 1, .., n$. A minterm c can also be defined as a BF for which there exists a unique choice of Boolean variables $x_1^*, .., x_n^* \in \{0, 1\}$ such that $c(x_1, .., x_n) = 1$ iff $x_1 = x_1^*, ..., x_n = x_n^*$. Hence, it follows that there are 2^n minterms. A DF of a BF f that is a sum of minterms is called a *disjunctive normal form* (DNF). It follows that the number of terms in a DNF is the number of *ones* in the truth table. The set of minterms that appear in the DNF of f is denoted by DNF(f).

It is often the case that the DNF can be reduced to a smaller number of terms. For example, the DNF $x_1x_2x_3\bar{x}_4 + x_1x_2x_3x_4$ can also be written as simply $x_1x_2x_3$. This is called the problem of DNF minimization. A reduction can be achieved by finding the so-called *prime implicants* of f. A product term c is called an implicant of f if the following statement holds: $c(x_1, .., x_n) = 1$ implies $f(x_1, .., x_n) = 1$. If the removal of any literal from an implicant c makes it a non-implicant, then c is called a *prime implicant*. We denote the set of prime implicant product terms of f by PI(f). The sum of all prime implicants is called the Blake canonical form. It can be reduced further by determining the *essential* prime implicants. A *minimized disjunctive form* (MDF) is a DF in which the number of terms is minimal. Computing an MDF can be handled by the Quine–McCluskey algorithm which can be used for a small number of inputs. For a larger number of inputs, the Espresso heuristic minimizer is the most

popular algorithm [163]. Details of these algorithms can be found in [192].

A Boolean function which maps every element from $\{0,1\}^n$ to a value of 0 or 1 is called completely specified. For purposes of the exact synthesis algorithm to be discussed, it is also useful to introduce the notion of an "incompletely specified" or *partial* Boolean function, which maps a (possibly proper) subset of the elements from $\{0,1\}^n$ to a value of 0 or 1. An incompletely specified Boolean function f on n variables is a mapping $f : D \rightarrow \{0,1\}$ where $D \subset \{0,1\}^n$. The set $\{0,1\}^n \setminus D$ gives the don't-care conditions of the function f.

A (partial) Boolean function can be described using three sets: the on-set, off-set and don't-care set. These three sets form a partition of the domain $\{0,1\}^n$ and can be used to fully describe the function in terms of the entire input space. The on-set of a Boolean function f, denoted $\textsc{on}(f)$, is the subset of $\{0,1\}^n$ which f maps to 1. The off-set of a Boolean function f, denoted $\textsc{off}(f)$, is the subset of $\{0,1\}^n$ which f maps to 0. The don't-care set of a Boolean function f, denoted $DC(f)$, is the subset of $\{0,1\}^n$ which f neither maps to 1 nor 0. It is the set $\{0,1\}^n \setminus D$. Since the on-, off-, and don't-care sets form a partition of the input space, only two of the three sets are required in order to describe the function. Using the on- and off-set notation we can distinguish between completely-specified and incompletely-specified functions. A completely-specified function has an empty don't-care set while the don't-care set of an incompletely specified function is non-empty.

If L and U are Boolean functions, then the interval $[L,U]$ is the set of functions defined by $[L,U] = \{f \in B : L \leq f \leq U\}$, where B is set of all n-variable completely specified Boolean functions. L is the lower bound of the interval, while U is the upper bound of the interval. The requirement $L \leq f$ implies that every function f contained in the interval must evaluate to 1 on the same set of input combinations on which L evaluates to 1. The requirement $f \leq U$ implies that every function f in the interval must evaluate to 0 for the same set of input combinations on which U evaluates to 0. Intervals can be used to describe incompletely specified functions. The interval $[L,U]$ represents the Boolean function that evaluates to 1 on the same set of input combinations as L and evaluates to 0 on the same set of input combinations as

U. The incompletely specified function function f can be defined as $\left[\lfloor f \rfloor, \lceil f \rceil\right]$. The relationships between the upper and lower bounds of a Boolean function f with the on-, off-, and don't-care sets of f are as follows:

$$\text{ON}(f) = \lfloor f \rfloor \qquad \text{OFF}(f) = \overline{\lceil f \rceil} \qquad DC(f) = \overline{\lfloor f \rfloor} \cdot \lceil f \rceil$$

5.2.2 Formal definition of the problem

An (N, q)-network is a realization of an n-input BF mappings $f_1, .., f_p$, represented as a directed acyclic bipartite graph (also known as a acyclic Petri net [193]), for which:

1. "places" correspond to cells (the output node is a cell with no out-edges);
2. "transitions" correspond to *QS nodes* or *input nodes*. The input node has no in-edges.

Assume there are m cells, each cell i computes a BF $c_i, i = 1, .., m$ that is realizable by a maximum of N NOR2 gates. A QS node then computes an OR gate of these outputs.

Definition 1. *Let $f_j : \{0,1\}^n \to \{0,1\}, j = 1, .., p$ be BFs. The vector BF $F := (f_1, .., f_p)$ is said to be (N, q)-realizable if there exists an (N, q)-network, with m cells and q QS nodes, and BFs $c_1, ..., c_m$, each of which is realizable by at most N NOR2 gates. The function F is evaluated by the network as follows: each cell computes the corresponding function c_i and each QS node computes an OR operation. For each input $w \in \{0,1\}^n$, w_i is substituted into the ith input node of the network and it is used to compute values for all cells and QS nodes. The value of the jth output node is taken as the value of $f_j(w)$.*

Remark 1. *Instead of distinguishing between the two types of node: cells and QS, an equivalent graphical description is in terms of directed hypergraphs. A directed hypergraph is specified by a set of nodes, which can be taken as the cells, and a set of "hyperedges", i.e. pairs (S_i, T_i) where each of S_i and T_i is a subset of the set of cells, thought of as the "source" and "target" of the "hyperedge" in question. Each QS node*

may be replaced by a hyperedge that has as its source the in-edges to QS and as its target the out-edges of QS. However, it is intuitively preferrable to use explicitly the two types of nodes as each of them computes a Boolean function in our application. Physically, one may think of QS nodes as concentrations of DSMs in the environment of the cell population, making the mapping into physical variables more transparent.

5.2.3 Exact synthesis algorithm for $(N, 0)$-networks

The next step is to find the circuit representation using a minimum number of NOR2 gates for each cell in the design. The exact synthesis algorithm described here does this by systematically generating circuits in search of a one that uses the minimum number of NOR2 gates. The algorithm represents a group of circuits using a partial network. The circuits represented by a partial network are those that can be created by adding edges and possibly nodes to the partial network. The algorithm searches through the space of possible circuits by incrementally completing the partial network. The search space is divided by choices made when completing the partial network. This results in a branch-and-bound algorithm that recursively performs an exhaustive search of the space in order to guarantee the optimality of the solution. To make the exhaustive search more efficient, the algorithm makes use of pruning techniques to help minimize the search space.

Networks and Boolean Functions

For the purposes of the exact synthesis algorithm, a (Boolean) network consists of nodes and directed edges (which are the inputs and the outputs of the nodes). Each node is either a primary input to the network or it is a node associated with a Boolean function, NOR2 in our context, which is the function produced at the output of this node. The function can be expressed either in terms of its immediate inputs or in terms of the primary inputs of the network. The local function is the Boolean function associated with the NOR2 gate expressed in terms of its immediate inputs, while the global function is the function associated with the gate expressed in terms of the

primary inputs. A relationship exists between the global functions at the inputs and output of a gate node based on the local function of the node. The relationship for the NOR2 gate is as follows:

$$f = \bar{g} \cdot \bar{h}$$
$$\text{ON}(f) = \text{OFF}(g) \cdot \text{OFF}(h)$$
$$\text{OFF}(f) = \text{ON}(g) + \text{ON}(h)$$

During intermediate stages of the algorithm partial or incomplete networks will be created. A partial network can be viewed as representing a set of circuits, namely those that can be created by adding edges and possibly nodes to the partial network. There are two types of gate nodes in a partial network, bounded and unbounded nodes. A node of a partial network is bounded if the size of the fan-in set has reached 2, the fan-in restriction imposed by using NOR2. A node is unbounded if the size of the fan-in set is less than 2. The Boolean function of a gate node will be determined by inputs to the gate. However, since the network is incomplete, there is a possibility that some nodes may be unbounded. The unassigned inputs of an unbounded gate node can take any function. Therefore when computing the global function of an unbounded node, the unassigned inputs are given the function $[0, 1]$ ($\text{ON} = 0, \text{OFF} = 0$).

The algorithm moves through the search space by adding nodes and edges to the partial network. The addition of an edge or a node to a partial network will have an impact on the function associated with the nodes on either side of this edge or node. In addition, the change in the function of a node may have an impact on the function of the nodes in its fan-in and fan-out sets. The relationships between the inputs and outputs of the NOR2 gate given above will be used to determine the implications that result when changes are made in a partial network. These implications will keep the global functions at the inputs and outputs of each gate node consistent with the

local function described by the logic gate.

$$f = \bar{g} \cdot \bar{h}$$

$$\textsc{on}(f) \rightarrow \textsc{off}(g) \quad \text{and} \quad \textsc{on}(f) \rightarrow \textsc{off}(h)$$

$$\textsc{on}(g) \rightarrow \textsc{off}(f) \quad \text{and} \quad \textsc{on}(h) \rightarrow \textsc{off}(f)$$

$$\textsc{off}(g) \cdot \textsc{off}(h) \rightarrow \textsc{on}(f)$$

$$\textsc{off}(f) \cdot \textsc{off}(g) \rightarrow \textsc{on}(h)$$

$$\textsc{off}(f) \cdot \textsc{off}(h) \rightarrow \textsc{on}(g)$$

Connectible Nodes

There are two sets of constraints which determine what edges can be added to a partial network. These constraints are based on the functional and structural properties of the network. The functional constraints determine which nodes can be connected to the input of a given gate node based on the global functions of the nodes. Given the current function at a node N and the functions at its inputs, a constraint function can be created which when solved will give the permissible functions for the node's inputs. Any node whose Boolean function falls within this set of permissible functions will be considered functionally connectible to the node N. The first step is to find the constraints for the set of permissible functions. For a NOR2 node, a function g is permissible as an input to a node if $f = \bar{g} \cdot \bar{h}$ where f is the function at the output of the node and h is the function at one of the inputs to the node. If f and h are completely specified functions, then any function g that satisfies this constraint would be considered a permissible function as an input to the node. However, the functions f and h are not always completely specified.

The constraint for determining the permissible functions must allow for incompletely specified functions at both f and h. A function g is permissible as an input to a node with output function $[\lfloor f \rfloor, \lceil f \rceil]$ and input function $[\lfloor h \rfloor, \lceil h \rceil]$ if for some function $f \in [\lfloor f \rfloor, \lceil f \rceil]$ and some function $h \in [\lfloor h \rfloor, \lceil h \rceil]$, $f = \bar{g} \cdot \bar{h}$. This can be

simplified into the following constraint.

$$\left(\lfloor f \rfloor \leq f \leq \lceil f \rceil\right) \cdot \left(\lfloor h \rfloor \leq h \leq \lceil h \rceil\right) \cdot \left(f = \bar{g} \cdot \bar{h}\right)$$

Any function g which satisfies this constraint is considered a permissible function to the node. There may be more than one function that satisfies this constraint. The interval describing the set of permissible functions can be found directly after some transformations on this constraint.

$$\lceil g \rceil = \overline{\lfloor f \rfloor} \qquad \lfloor g \rfloor = \overline{\lceil f \rceil} \cdot \overline{\lceil h \rceil}$$

A node C in a network is functionally connectible to a gate node N if the intersection of C's function interval with the interval of permissible functions for an input of N is non-empty. The intersection of the two intervals gives the set of functions from the permissible set that the node C can take. Let $C \in \left[\lfloor C \rfloor, \lceil C \rceil\right]$ be the function at node C. C is functionally connectible to a gate node N with output function f, input function h, and permissible functions g if the following condition is satisfied:

$$\left[\lfloor C \rfloor, \lceil C \rceil\right] \cap \left[\lfloor g \rfloor, \lceil g \rceil\right] \neq \emptyset$$

This requirement can also be stated as:

$$\begin{aligned}
\lfloor C \rfloor \leq \lceil g \rceil \quad &\text{and} \quad \lceil C \rceil \geq \lfloor g \rfloor \qquad \text{or} \\
\lfloor C \rfloor \leq \overline{\lfloor f \rfloor} \quad &\text{and} \quad \lceil C \rceil \geq \overline{\lceil f \rceil} \cdot \overline{\lceil h \rceil} \quad \text{or} \\
\textsc{on}(C) \cdot \textsc{on}(f) = 0 \quad &\text{and} \quad \textsc{off}(C) \cdot \textsc{off}(f) \cdot \textsc{off}(h) = 0
\end{aligned}$$

The second set of constraints that must be satisfied when adding edges to the network are structural constraints. These constraints are needed to insure that the network remains acyclic. A node C is structurally connectible to a node N if C does not appear on any path from N to a primary output. In other words, C can not be an ancestor of N.

The connectible set of a node N is the set of nodes in the partial network that are both functionally and structurally connectible to N.

Covering

The notion of covering is central to the exact synthesis algorithm. A minterm in the off-set of the global function of a NOR2 gate node N is covered if the minterm appears in the on-set of the global function of at least one of N's fan-in nodes. Using this notion of covering, the off-set minterms of a NOR2 gate can be divided into two sets. The covered set is the set of minterms that are already covered by a fan-in of the node: $\textsc{off}(f) \cdot \big(\textsc{on}(g) + \textsc{on}(h)\big)$, where f is the global function describing the output of the node and g and h are the global functions describing the two inputs to the node. The uncovered set is the set of minterms that are not yet covered by the inputs of the gate node: $\textsc{off}(f) \cdot \overline{\textsc{on}(g)} \cdot \overline{\textsc{on}(h)}$. When the uncovered set becomes empty, the node is completely implemented by its inputs. These nodes are called covered. The algorithm will perform a covering when the fan-in set of a node is changed or the global function of one of the fan-in nodes is changed so that at least one minterm from the node's uncovered set moves to its covered set.

Algorithm Description

The branch-and-bound algorithm will explore the search space recursively. Each step of the recursion will perform a covering of at least one uncovered minterm from some node in the partial network. When the current network is determined to be complete or its cost becomes larger than the current cost bound, the recursion stops and backtracks to the previous step.

The algorithm is initially called on a partial network that represents the entire set of circuits in the search space. This initial network will contain one gate node for each output function and one input node for each primary input. The fan-in and fan-out sets of all these nodes are empty, while the global function at each node is set according to the function it represents. In addition, the connectible set for each of the gate nodes must be computed according to the constraints described above.

The `ExploreNetwork` procedure is the main procedure of the algorithm. It takes as input a single partial network. It performs a covering of some uncovered minterm in the partial network and then recursively calls the same procedure using the new partial network. The pseudocode for the `ExploreNetwork` procedure is given in below where Net, M, and N are shorthand for network, minterm and node, respectively.

Algorithm 1: Exact Synthesis Algorithm

```
ExploreNetwork(Net):
    if AllCovered(Net) then
        Store Net as optimal solution
        UpperBound ← Net.cost
    else
        M, N ← SelectMintermForCovering(Net)
        Connectible ← FindConnectibleSet(N, M)
        for every C ∈ Connectible do
            NewNet ← PerformCovering(Net, N, C, M)
            if NewNet.cost < UpperBound then
                ExploreNetwork(NewNet)
            end
        end
    end
```

The `ExploreNetwork` procedure begins by first determining whether every gate node in the network is covered by its inputs. This is performed by the `AllCovered` function. A gate node G is covered by its inputs I_1 and I_2 if $\text{OFF}(G) = \text{ON}(I_1) + \text{ON}(I_2)$. If every gate node is completely covered,the network is a complete circuit. Therefore it can be stored as the current optimal network and the cost bound can be set accordingly.

If there are nodes remaining in the network that are not fully covered by their inputs, then work still remains towards completing this partial network. The procedure will attempt to cover an uncovered minterm from at least one node. The covering process begins by first selecting an uncovered minterm and its corresponding node from the network. This is done by the `SelectMintermForCovering` procedure. The minterm that is chosen by this procedure is guaranteed to be covered during this step of the recursion. It may be the case that additional uncovered minterms in the

network may be covered as well. This will depend on the covering that is made.

The procedure continues from the minterm selection to obtaining the connectible set for the node whose off-set minterm is to be covered. A connectible set is stored for every node. This set gives all the possible ways that exist in the network for covering every uncovered minterms from the node. The procedure `FindConnectibleSet` will prune the node's connectible set down to only those nodes that can cover the selected minterm. In order for a node C in N's connectible set to be included in *Connectible*, the minterm M must not appear in the off-set of C. C is added to *Connectible* if $\text{OFF}(C) \cdot M = 0$. A node C satisfying this condition can be used to cover M since M must be contained in (or moved to) C's on-set in order for C to cover M. The `FindConnectibleSet` procedure will also add a new gate node to the set *Connectible* if the fan-in set of N is less than 2. This gate node will be a new node which does not yet appear in the network.

Each node in the connectible set returned by the `FindConnectibleSet` procedure must be considered as a way of covering the selected minterm. The enumeration of the possible coverings using each node in the connectible set corresponds to the branching part of the branch-and-bound search. For a single connectible node, C, a new network is created by the `PerformCovering` procedure. This new network will be a supernetwork of *Net* with the added covering of M using C.

The process of covering M can add a connection between the nodes N and C as well as change the global function at C. Based on the relationship between the global functions at a node's fan-in and fan-out described by the node's local function, the changes made at N and C can effect the global functions at other nodes in the network. The changes to the global functions that result from a covering are called functional implications of the covering. These functional implications must be completed to maintain the consistency of the global functions.

Once the functional implications have been completed, the `PerformCovering` procedure updates the connectible set of each node in the network. Since the global function may have changed for some nodes, the set of nodes that are now connectible may have also changed. It is possible that previously covered nodes have now become

uncovered, and therefore the set of connectible nodes will need to be computed for these nodes as well. The connectible set of each uncovered gate node in the network can be found according to the constraints described above. The connectible set update will conclude the `PerformCovering` procedure.

Once the covering is complete, the cost of the partial network is compared to the current cost bound. If the cost of the new partial network is greater than the current cost bound, this network can be pruned. This is the case since each elementary network in the set that the new network represents will have cost larger than the current cost bound. Therefore, none of the elementary networks represented can be optimal. If the cost of the new network is less than or equal to the cost bound then it is possible that some elementary networks in the set that this partial network represents may have cost less than or equal to the current cost bound. Thus requiring this portion of the search space to be explored further. The portion of the search space represented by this new partial network is explored by calling the `ExploreNetwork` procedure on the new network.

When the recursive call returns, the next node in the connectible set is used to cover the selected minterm. Once again the search space represented by the new partial network is explored. Each possible way of covering the minterm is selected and then explored until all options have been exhausted. Once the recursion returns for the last time, all complete networks represented by the original partial network have been explored. Any optimal networks represented by the original partial network have been stored as such. Now the procedure can return to the previous recursive call since this call has been completed. This ends our description of the exact synthsis algorithm.

5.2.4 Minterms: realizable and offending

Our framework proposed in Fig. 5.2 relies on representing a BF as a disjunction of other BFs. Hence, we first study the DNF in terms of minterms. We provide the following definition:

Definition 2. *Let N be a given positive integer. A minterm in n variables is said to be offending with respect to N if it cannot be realized with N NOR2 gates or less. The set of all offending minterms with respect to N is denoted as $\mathcal{O}_N$.*

The following statement that can be proven using various methods. It can also be verified via the exact synthesis algorithm which we have introduced earlier for any given n.

Theorem 1. *Let c be a minterm of n variables with $n \geq 2$. Then the minimum number of NOR2 gates to realize c is $3(n-1) - q(c)$, where $q(c)$ is the number of complemented literals in c*

Example. The minterm x_1x_2 needs $(3)(1) - 0 = 3$ gates, while the minterm $x_1x_2\bar{x}_3$ needs (3)(2)-1=5 gates.

It follows that any minterm of *three* variables or less can be realized with six gates or less. Therefore, the design in Fig. 5.2 can realized *any DF* of any BF f of three variables or less.

For minterms with four variables, it follows that a minterm with four or three uncomplemented literals cannot be realized with 7 NOR gates. The minterm $x_1x_2x_3x_4$ needs at least 9 NOR gates, and the minterms $\bar{x}_1x_2x_3x_4$, $x_1\bar{x}_2x_3x_4$, $x_1x_2\bar{x}_3x_4$, $x_1x_2x_3\bar{x}_4$ need at least eight NOR gates. These five minterms are called the *offending minterms* with respect to $N = 7$ gates. If an MDF contains one or more offending minterms, additional DSMs are used to divide up these circuits further.

5.2.5 Lower bound on the number of $(N, 1)$-realizable BFs

We state the following Theorem:

Theorem 2. *Let f be a BF, and let N be given. If $\mathcal{O}_N \cap PI(f) = \emptyset$, then f is $(N, 1)$-realizable.*

Proof. Let us write f as an MDF: $f(x_1, .., x_n) = \sum_{i=1}^{m} c_i(x_1, .., x_n)$. By the assumption, $c_i \notin \mathcal{O}_N$ for all i. Hence, f can be realized using the disjunctive form design in Fig. 5.2 where each c_i is $(N, 0)$ realizable. □

In other words, any function that does not have a prime implicant offending minterm is $(N, 1)$-realizable. Hence, our objective in this subsection is to count those BFs combinatorially.

In order to facilitate the discussion, a binary representation of minterms is used. An n-variable minterm can be represented with n bits. If a literal is uncomplemented then it is represented as 1, otherwise it is represented as 0. For instance, the minterm $x_1x_2x_3\bar{x}_4$ is written as 1110. Using this notation, the *distance* between two minterms is defined as the *Hamming distance* between their binary representations. Two minterms are *neighbors* if their distance is 1. Hence, we are ready to state the following result:

Lemma 3. *Let f be a BF, and let $c \in DNF(f)$ be a minterm. If there exists a minterm $c^* \in DNF(f)$ such that c, c^* are neighbours then c, c^* are not prime implicants.*

Proof. Since both $c, c^* \in \mathrm{DNF}(f)$, then $c + c^*$ appears in the DNF of f. But since c, c^* differ in only one literal, then $c + c^*$ simplifies into a product term with a smaller number of literals. The resulting product term is an implicant of f, hence c, c^* are not prime implicants. □

The following test follows:

Theorem 4. *A BF f is $(N, 1)$-realizable if for every $c \in \mathcal{O}_N \cap DNF(f)$, there exists $c^* \in DNF(f)$ such that c, c^* are neighbors.*

Hence, our task reduces to counting BFs that have a minterm in $\mathcal{O}_N \cap \mathrm{DNF}(f)$ without neighbours. We propose a fast algorithm to find the set of BFs that satisfy the conditions of Theorem 4. We use the notation $\mathcal{N}(c)$ as the set of neighbors of a minterm c. The algorithm is given below:

Parameters: n number of variables, N the maximum number of gates per cell.

Initialization: Set $X = 0$;

for $i = 0 : 2^n - 1$ **do**

flag$_1$:=0;

for $c \in \mathcal{O}_N$ **do**

flag$_2$:=1;

for $c^* \in \mathcal{N}(c)$ **do**

if c^* is an implicant of f, **then** flag_2=0; break; **end**

end

flag_1:=flag_1+flag_2;

end

if $\text{flag}_1 \neq 0$, **then** $X := X + 1$; **end**

end

Output: X is an upper-bound on the number of networks which are *not* $(N, 1)$-realizable.

Realization with offending minterms: Proof of Theorem 5

We provide a constructive proof of Theorem 5. If $\mathcal{O}_N \cap \bigcap_{j=1}^{p} S_N = \emptyset$, then the numbers of DSMs needed is p at most. Next, we study the cases of more than one offending minterm. Remember that $\mathcal{O}_7 = \{1111, 1110, 1101, 1011, 0111\}$ and $\mathcal{O}_8 = \{1111\}$. For the $N = 9$, we have $\mathcal{O}_9 = \emptyset$, and hence the corresponding statement follows directly.

Vector BFs with one offending minterm We study the case of only one offending minterm appearing as a prime implicant. For instance, consider the minterm 1111 which is offending for both the cases $N = 7$ and $N = 8$.

Let $x \boxplus y := \text{NOR}(x, y)$, and remember that $\bar{x} = x \boxplus x$. A minimal realization of the minterm 1111 consists of 9 gates and is given as follows:

$$x_1x_2x_3x_4 = \bar{x}_4 \boxplus \overline{\bar{x}_3 \boxplus \overline{\bar{x}_2 \boxplus \bar{x}_1}}.$$

One DSM molecule can be used to partition the realization above into two cells with four and five gates as follows:

$$x_1x_2x_3x_4 = \bar{x}_4 \boxplus \overline{\bar{x}_3 \boxplus z},$$
$$z = \overline{\bar{x}_1 \boxplus \bar{x}_2},$$

where z is communicated via a DSM. In function format, we are writing $x_1x_2x_3x_4 = f_1(f_2(x_1, x_2), x_3, x_4)$ where f_2 is $(N, 0)$-realizable and f_1 is $(N, 1)$ realizable, where $N \geq 5$. This partition is depicted in Fig. 5.3-b). Note that this completes the proof of the statement of Theorem 5 for the case $N = 8$.

For the case $N = 7$. The other four offending minterms can be treated similarly. Their realizations requiring 8 NOR gates can be partitioned into two 4-gate cells.

BFs with two offending minterms For any two minterms chosen, they will share exactly two uncomplemented literals. For instance consider $1110, 1101$. Both x_1, x_2 are uncomplemented. Then, the following realization is proposed:

$$x_1x_2x_3\bar{x}_4 = x_4 \boxplus \overline{\bar{x}_3 \boxplus z},$$
$$x_1x_2\bar{x}_3x_4 = \bar{x}_4 \boxplus \overline{x_3 \boxplus z},$$
$$z = \overline{\bar{x}_1 \boxplus \bar{x}_2},$$

where z is communicated via a DSM. Hence, we compute x_1x_2, and then use it for computing the full minterm. This partition is depicted in Fig. 5.3-a).

Hence, any vector BF with no more than two offending minterms needs no more than $p + 1$ DSMs.

BFs with three or more offending minterms If three or more offending minterms in the set $\{1110, 1101, 1011, 0111, 1111\}$ appear in the MDFs of a vector BF then we will show below that no more 2 DSMs are needed.

Let's consider the case of five offending minterms. It can be seen that this case can be realized with two DSMs as follows. The four minterms can be divided in no particular order into two subsets which each share exactly two uncomplemented literals. Then, a similar design follows.

$$x_1x_2x_3x_4 = \bar{x}_4 \boxplus \overline{\bar{x}_3 \boxplus z_1},$$
$$x_1x_2x_3\bar{x}_4 = x_4 \boxplus \overline{\bar{x}_3 \boxplus z_1},$$

$$x_1x_2\bar{x}_3x_4 = \bar{x}_4 \boxplus \overline{x_3 \boxplus z_1},$$
$$x_1\bar{x}_2x_3x_4 = x_1 \boxplus \overline{\bar{x}_2 \boxplus z_2},$$
$$\bar{x}_1x_2x_3x_4 = \bar{x}_1 \boxplus \overline{x_2 \boxplus z_2},$$
$$z_1 = \overline{\bar{x}_1 \boxplus \bar{x}_2},$$
$$z_2 = \overline{\bar{x}_3 \boxplus \bar{x}_4},$$

where z_1 and z_2 are communicated via two different DSMs. Hence, no more than $p + 2$ DSMs are needed.

5.2.6 Counting $(7, 2)$ and $(7, 3)$ realizable BFs

In the case of a scalar BF, we can count the number of BFs realizable with the disjunctive form design combinatorially. For a single offending minterm. We note that if $1111 \in \mathrm{DNF}(f)$, then the rest of the offending minterms cannot appear, otherwise, they will be neighbours, and the associated BF will be (7,1)-realizable. This amounts to specifying a fixed value to the vector ($f(1, 1, 1, 1)$, $f(1, 1, 1, 0)$, $f(1, 1, 0, 1)$, $f(1, 0, 1, 1)$, $f(0, 1, 1, 1)$), namely $(1, 0, 0, 0, 0)$ out of 2^5 possible choices. The total number of these BFs is $2^{16}/2^5 = 2^{11} = 2048$. The other four minterms need be handled together since they share neighbors. Similar combinatorial computations gives the total number of BFs with one offending minterm to 7,808.

If two offending minterms in the set $\{1110, 1101, 1011, 0111\}$ appear in the DNF without neighbours then these have been shown to be $(7, 2)$-realizable. The total number of BFs with exactly two offending minterms is 960.

The total number of BFs with exactly three offending minterms is $2^7 = 128$ BFs. For the case of a scalar BF we can do better than the upper bound given in Theorem 5 since any three offending minterms can be realized with one DSM as follows:

$$c(x_1, x_2, x_3, x_4) = \bar{x}_1x_2x_3x_4 + x_1\bar{x}_2x_3x_4 + x_1x_2\bar{x}_3x_4 = zx_4,$$

where $z = \bar{x}_1x_2x_3 + x_1\bar{x}_2x_3 + x_1x_2\bar{x}_3$ can be computed via a single DSM.

Hence this expands the set $(7, 2)$ scalar BFs to 8896 using the disjunctive design. Finally, there are 2^5=32 BFs with four offending minterms.

5.2.7 A graph partitioning algorithm

This algorithm is implemented by first expanding the design provided by the exact synthesis algorithm so that the output of each NOR2 gate is only used once, effectively creating repeated 'branches'. For each gate in the expanded design, the total number of gates is computed such that this number accounts for all the gates located downstream of a given gate plus the gate itself. To determine that a design can be partitioned using 1 DSM, we first look at the total number of gates of the upstream node T_{up}. Going through the other gates, there must exist one gate i such that its total number of gates T_i with n occurences within the expanded design that fulfills the following conditions:

$$T_{up} - nT_i \leq N,\ T_i \leq N.$$

In the case of partitioning using 2 DSMs, two gates i and j are used. There are two possibilities of partitioning, one in which neither i or j is located downstream (or upstream) of the other gate. In this case, the conditions are simply that:

$$T_{up} - n_i T_i - n_j T_j \leq N,$$

$$T_i \leq N, T_j \leq N.$$

The second scenario occurs when gate i is either located upstream or downstream of j. By assuming that we order i and j such that i is the upstream gate, the conditions to be realizable with 2 DSMs become:

$$T_{up} - n_i(T_i - T_j) - n_j T_j \leq N,$$

$$T_i - T_j \leq N, T_j \leq N.$$

5.3 Results

Current synthetic biology applications, from the detection of environmental toxins to the design of engineered immune cells, involve typically only a handful of inputs. Thus, even though the methods that we introduce can be, in principle, scaled to an arbitrary number n of inputs, for computational tractability we will we restrict our attention to BFs of 4 inputs or less. Note that there are 2^{2^n} BFs with n inputs. This is because there are 2^4 possible binary strings of 4 bits $(0000, 0001, \ldots, 1111)$, and each of these can be mapped into either a 0 or a 1. In particular, there are 2^{16} possible Boolean functions with 4 inputs.

In the following sections, we discuss the number of functions that can be realized with no DSMs (so that a single colony is sufficient), and then proceed to the topic of this paper proper, namely larger numbers of DSMs.

5.3.1 $(N, 0)$-realizable BFs via exact synthesis

As a first step, BFs that are realizable without any DSMs are considered. We use a branch-and-bound algorithm (see Methods) to find the minimal number of gates required to realize a given BF. The results are summarized in Fig. 5.1. In particular, we find that only about 11.69% of the 2^{16} different BFs with 4 inputs can be realized using seven NOR gates or less. Moreover, we find that 4-input BFs can require up to fourteen NOR gates, and they most frequently require ten NOR gates, with a median of 10. Thus, most realizations are not implementable in a single cell via the current technological limitation of around seven gates per cell.

The algorithm was used to generate the data for the histograms in Fig. 5.1, but the main goal was to populate a database that provides optimal (minimal number of gates) realizations for all 4-input Boolean functions. This database is used subsequently as a lookup table, to assign an optimal NOR2 design per cell when building distributed designs.

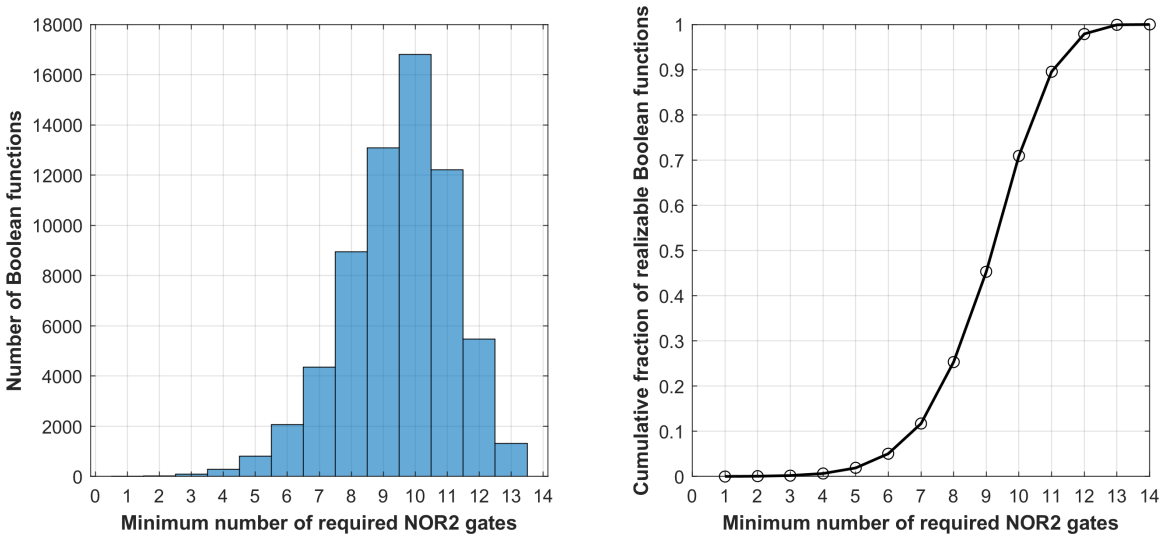


Figure 5.1: The minimal number of gates required to represent a given 4-input Boolean function. This assumes no use of DSMs, or in the language of synthetic biology, circuits that can be built within one cell. (a) Total number of BFs vs. minimal number of gates needed for realization, (b) the cumulative fraction of realizable Boolean functions vs. the number of NOR2 gates.

5.3.2 $(N, 1)$-realizable BFs via disjunctive form design

The main focus of this paper is to study the case when a circuit design cannot readily fit in a cell (or equivalently, it is not $(N, 0)$-realizable *in the terminology introduced before*). We first propose a bottom-up approach in which a Boolean realization is built "from scratch" in a distributed manner. We call this approach *Disjunctive Form Design (DFD)*. We then present a top-down approach in which, starting from a circuit realizing a given BF taken from the pre-computed database, we apply a graph partitioning algorithm to distribute the computation.

We introduce DFD next. First, we study the case when a *single DSM* can be used to distribute the computation. We propose a design depicted in Fig. 5.2 to realize a disjunctive form of f. Each cell releases the same DSM. The total concentration of the DSM adds up, hence its concentration acts as an OR gate. Hence, BFs $c_1, .., c_m$ are to be found such that

$$f(x_1, .., x_n) = \sum_{j=1}^{m} c_i(x_1, .., x_n). \tag{5.2}$$

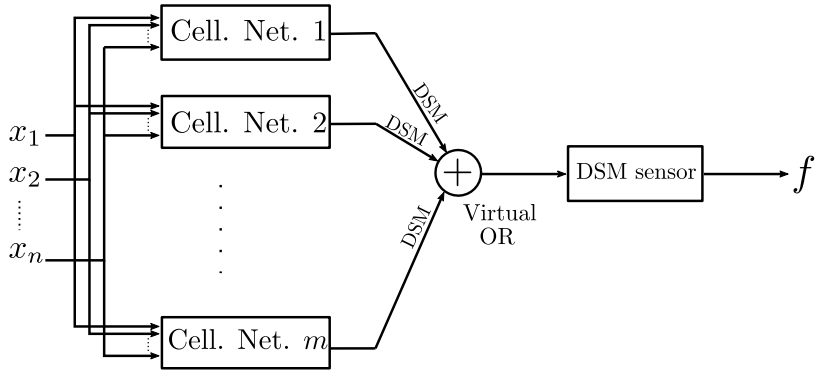


Figure 5.2: Disjunctive form design. The abbreviation "Cell. Net." stands for "Cellular Network". The figure depicts a distributed computation design for implementing a Boolean function f of n Boolean variables with one DSM. The DSM sensor can be, for instance, a DSM-to-GFP cell. A cellular network can consist of multiple cells communicating with DSMs. If each cellular network consists of a single cell, and each cell does not contain more than N gates, then the design is an $(N, 1)$-realization.

We interpret the summation in the Boolean sense, as an OR gate (See Methods for more details). The summation in (5.2) uses one DSM. If each of the BFs $c_1, .., c_m$ can be realized with N gates or less then f is $(N, 1)$-realizable. Such a design implements some of the computation *(namely, the OR gate)* outside the cells, and this allows us to realize many of the BFs that cannot be realized via graph partitioning with a single cut to a full design drawn from the database (see Fig. 5.5 as an example).

The next step is to determine the circuit for each of the cellular networks in Fig. 5.2. This requires us to determine the corresponding BFs $c_1, .., c_m$. One method is to write f as a Minimized Disjunctive Form (MDF) which implies that $c_1, .., c_m$ are *product terms* (see Methods for definition). Hence, a basic scheme assigns each cell in Fig. 5.2 with the task of computing the corresponding product term in the MDF of f. If we assume that each cellular network cannot contain more than one cell, this scheme allows us to compute the number $(N, 1)$-realizable functions that can be written in the form (5.2) as we will show later.

The disjunctive form design described above is a basic scheme that can be improved in many cases. This is since assigning one cell for each term of the MDF can be inefficient in other aspects. For example, when applied to the BF $\bar{x}_1 + \bar{x}_4 + \bar{x}_2 x_3 x_4$, the aforementioned scheme will provide a design consisting of three types of cells,

two implemented by just one gate each and the other one by six gates. While giving a $(6, 1)$ realization that satisfies the mandatory bounds, the imbalance between the computational resources allocated to each of the three types of cells may be undesirable in some applications and it can be improved (for instance) by reducing the number of cells. In general, it is possible to find several solutions satisfying the same bounds. To address the non-uniqueness of solutions, we will propose later a framework in which a user is able to choose the best solution among different trade-offs between N and q by defining secondary criteria.

Despite the drawbacks mentioned above of the basic scheme (i.e., the scheme of one cell for each term in the MDF), this procedure produces a *modular* design since *minterm-computing cells* (MCCs) can be designed once and reused as needed in the design of a single or multiple BFs. For instance, a four-input function needs only 13 types of such cells to compute minterms of up to 4 variables. For instance, $x_1x_2\bar{x}_3x_4$ and $x_1x_2x_3\bar{x}_4$ can be computed by the same circuit after permuting the inputs.

Realizability test. The next question that we tackle is that of finding the number of BFs that can be realized with DFD with only *one DSM*. We provide next a fast test that determines a lower bound on the number of $(N, 1)$-realizable BFs. It relies on the development of the concept of *offending minterms*, which are minterms that cannot be realized with N gates or less. For example, the minterm $x_1x_2x_3x_4$ (a 4-input AND) can be realized with at least 9 NOR2 gates, and hence it is not $(7, 0)$-realizable. To that end, we use $\mathcal{O}_N$ to denote the set of all such minterms. For instance,

$$\mathcal{O}_7 = \{x_1x_2x_3x_4\,,\ \bar{x}_1x_2x_3x_4\,,\ x_1\bar{x}_2x_3x_4\,,\ x_1x_2\bar{x}_3x_4\,,\ x_1x_2x_3\bar{x}_4\}.$$

After computing the set $\mathcal{O}_N$, a combinatorial test is provided, where we show that a BF f is $(N, 1)$-realizable if the set of prime implicants of f does not contain an offending minterm. (See Methods for further details)

In the case of $N = 7$, the test shows that there are at least $56,608$ (7,1)- realizable 4-input BFs out of a total of $2^{16} = 65,536$ BFs. Therefore, the percentage of realizable

BFs is at least 86.384% when using one DSM, compared to only 11.690 % without any DSMs for 4-input BFs, which amounts to a 7.39-fold increase. In the case of $N = 8$, the percentage of (8,1)-realizable functions is at least 96.877%, compared to 25.321% for (8,0)-realizable functions. Note that these bounds are not tight, as we will show in the next subsection.

5.3.3 More DSMs are needed: realization with offending minterms

We have shown that using just one DSM considerably extends the number of realizable BF. Still, there are BFs which cannot be realized with a single DSM. In particular, one DSM is insufficient if there are offending minterms in the minimized disjunctive form. In this subsection, we generalize the method in order to handle the case of offending minterms appearing in the MDF of a BF. This will be achieved by allowing some of the cellular networks in Fig. 5.2 to contain more than a single cell.

First, we propose a generic constructive method that aims at providing an upper bound on the number of DSMs needed to realize p BFs simultaneously (i.e., vector BFs). We summarize here the results for the cases of $N = 7, 8, 9$ (which are the given bounds on the number of gates per cell).

Theorem 5. *Given a 4-input vector BF $F = (f_1, .., f_p)$. Then:*

1. *For $N = 7$, there exists $0 \leq q \leq p + 2$ such that F is $(7, q)$-realizable (i.e., no more than $p + 2$ DSMs are needed).*

2. *For $N = 8$, there exists $0 \leq q \leq p + 1$ such that F is $(8, q)$-realizable (i.e., no more than $p + 1$ DSMs are needed).*

3. *For $N = 9$, there exists $0 \leq q \leq p$ such that F is $(9, q)$-realizable (i.e., no more than p DSMs are needed).*

A constructive proof is provided in the Methods section by relying on realizing the offending minterms individually by designing a corresponding cellular network. If an

offending minterm appears in multiple BFs, then it only needs to be computed once. For instance, consider the case when $x_1x_2x_3x_4$ is the only offending minterm that appears in the MDFs of $F = (f_1, .., f_p)$; then, its 9-gate realization can be partitioned via one DSM to produce a cellular network consisting of two cells whose number of gates just are 4 and 5 (Fig. 5.3-a). More offending minterms can be handled also. Consider, for example, the case when the only offending minterms are $x_1x_2x_3\bar{x}_4$, $x_1x_2\bar{x}_3x_4$. Then, the corresponding cellular network computes the common product term x_1x_2 first, and then the computed value is communicated to two other cells that compute the required minterms via a single DSM, resulting in a cellular network design with a total of 3 cells (with 4 gates each) and one DSM (Fig. 5.3-b). Other cases are described in the Methods.

Scalar BF with offending minterms. The results in Theorem 5 can be slightly improved in the case of a scalar BF and full databases of realizations with arbitrary q can be provided. Furthermore, We provide lower bounds on the percentage of realizable 4-input BFs via the disjunctive form design as shown in Table 5.1 (see Methods for details). Estimates are also provided for 5-input Boolean functions in Table 5.2.

(N, q)		q			
		0	1	2	3
N	7	11.690%	86.384%	99.951%	100%
	8	25.321%	96.877%	100%	100%
	9	45.297%	100%	100%	100%
	10	70.939%	100%	100%	100%
	11	89.570%	100%	100%	100%

Table 5.1: Percentage of realizable 4-input Boolean functions via the disjunctive form design (lower bounds). N refers to the maximum number of allowed NOR2 gates per cell, and q refers to the number of DSMs.

5.3.4 A graph partitioning algorithm

DFD represents a bottom-up approach to the problem of distributed computation. Here we study a top-down approach which can yield better results in some cases where

(N,q)		q					
		0	1	2	3	4	5
N	7	∼2%	11.0%	34.2%	60.6%	81.1%	93.0 %
	8	∼4%	42.3%	74.3%	90.4%	96.8%	99.00%
	9	∼10%	70.0%	92.4%	98.3%	99.6%	99.9%
	10	∼22%	88.6 %	98.6%	99.8%	99.8%	99.997%
	11	∼40%	98.4%	100%	100%	100%	100%
	12	∼63%	100%	100%	100%	100%	100%
	13	∼83%	100%	100%	100%	100%	100%

Table 5.2: Estimates of the realizable 5-input Boolean functions via the disjunctive form design (lower bounds). The $(N, 0)$ percentage values were estimated from the limited list of calculated optimal designs of 5-input Boolean functions provided in [1]. The $(N, q > 0)$ values were estimated by examining 100,000 MDF decompositions picked at random from the set of 2^{2^5} possible 5-input combinations. The reductions due to redundancy when dealing with more than 1 offending minterm were not accounted for, unlike the 4-input case. Thus, the listed estimations of the lower bounds are expected to be looser.

the number of DSMs needed can be reduced with respect to the DFD. We propose a graph partitioning algorithm applied to a full NOR2 network generated by the exact synthesis algorithm. In this formulation, the usage of a single DSM will be equivalent to a single cut in the extended NOR2 graph. See Methods for a detailed description of the algorithm.

Due to the nature of graph partitioning, unless the circuit readily fits inside a cell, the use of a DSM signal is necessary. Thus, for 4 inputs and $N = 7$, 11.690% of BFs can be realized without a DSM, another 37.680% of BFs can be realized using 1 DSM, and an additional 34.900% using 2 DSMs, using this technique. This leaves 15.730% of BFs that cannot be realized using 2 or less DSMs. Thus, given the fact that the number of mutually orthogonal DSMs is considered to be a limited resource, the disjunctive-form design is generally preferable as a way to minimize the use of DSMs. However, there exist cases for which the partitioning algorithm offers a more compact realization.

By combining both the disjunctive form design and graph partitioning, the numbers presented in Table 5.1 can be improved (for the case $q = 1$) to those shown in Table 5.3.

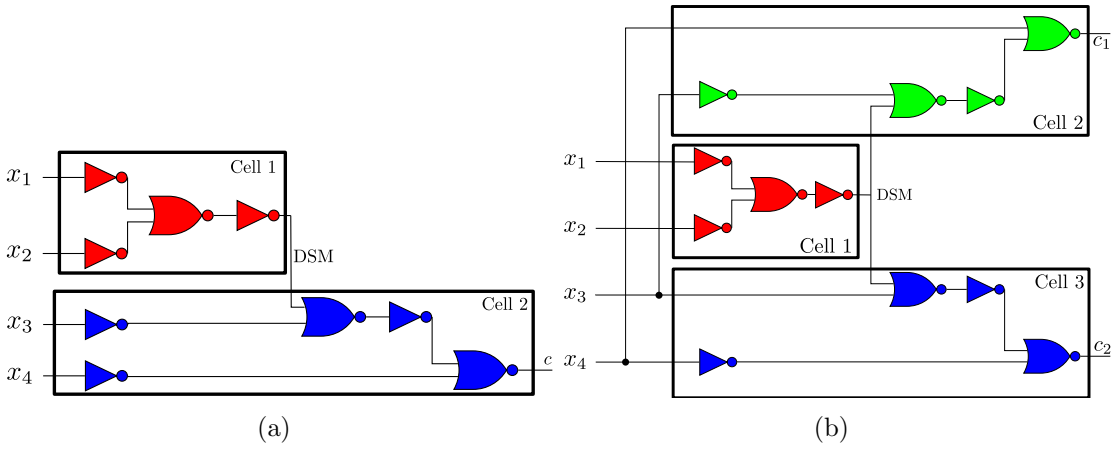


Figure 5.3: Construction of cellular networks to realize the offending minterms. Each diagram can be included as a cellular network in the DFD as shown in Fig. 5.2. (a) Partitioning a realization of the minterm $c = x_1x_2x_3x_4$ that requires a minimum of 9 NOR2 gates. Many partitions are possible. We show here one partition into two cells of 4 and 5 gates respectively, via the use of 1 DSM. (b) Partitioning a realization of two minterms sharing two literals $c_1 = x_1x_2x_3\bar{x}_4$, $c_2 = x_1x_2\bar{x}_3x_4$. These two minterms that require a minimum of 8 NOR2 gates each can be split up in such a way that the computation for the common product term x_1x_2 can be reused. This allows to use a total of three cells instead of four. Different colors indicate different cells.

5.3.5 Optimized distributed design framework

The designs presented so far provide a circuit representation that fulfills the physical constraints imposed (on numbers of gates and cell types). In the case of disjunctive form design, the basic scheme introduced earlier requires the use of at least as many cell types as there are minterms in the MDF of a given BF. Using the database developed through the exact synthesis algorithm, it is straightforward to test all the combinations of various non-offending product terms that are $(N, 0)$-realizable. This leads, in most cases, to a reduction in the number of required cells for a given realization. The graph partitioning algorithm provides also alternative designs in many cases. Hence, there is need to develop a framework to choose the "best" solution amongst the solutions available. In the parlance of optimization theory, we aim at creating a framework that enables the user to explore the "Pareto front" of solutions to choose from.

Generally, there is a trade-off between the maximal number of gates per cell (N)

(N, q)		q			
		0	1	2	3
N	7	11.690%	88.666%	99.951%	100%
	8	25.321%	97.440%	100%	100%
	9	45.297%	100%	100%	100%
	10	70.939%	100%	100%	100%
	11	89.570%	100%	100%	100%

Table 5.3: Lower bound on cumulative percentage of realizable 4-input Boolean functions combining the disjunctive-form design and graph partitioning algorithms

and the number of DSMs (q). Furthermore, depending on the application, secondary optimization criteria can be used depending on the application. For instance, the most intuitive criterion is to use as few cells and NOR gates as possible. In other cases, one may want to minimize the number of cells required, while also minimizing the variability in the number of NOR gates used per cell, in order to reduce differences in computational time between cells.

In the context of the DFD, the use of a look-up database allows for a rapid and exhaustive search of these optimal designs. This is possible by taking all the non-offending minterms and testing for all possible combinations of a given set of minterms. For example, the five possible combinations of the set $\{1, 2, 3\}$ are $\{\{1, 2\}, 3\}$, $\{1, 2, 3\}$, $\{\{1, 3\}, 2\}$, $\{1, \{2, 3\}\}$, and $\{\{1\}, \{2\}, \{3\}\}$. In this case, there are five different ways of combining 3 minterms. Given the use of the database, the computation required to compute all five designs is very minimal. This allows a rapid look-up of what minterms can be efficiently combined, while also meeting the constraints of the problem.

Finally, the partitioning algorithm can also be combined with the distributed disjunctive-form algorithm through combining offending and non-offending minterms to find combinations that lead to more compact or desirable designs. The complete framework is summarized in the flowchart depicted in Fig. 5.4.

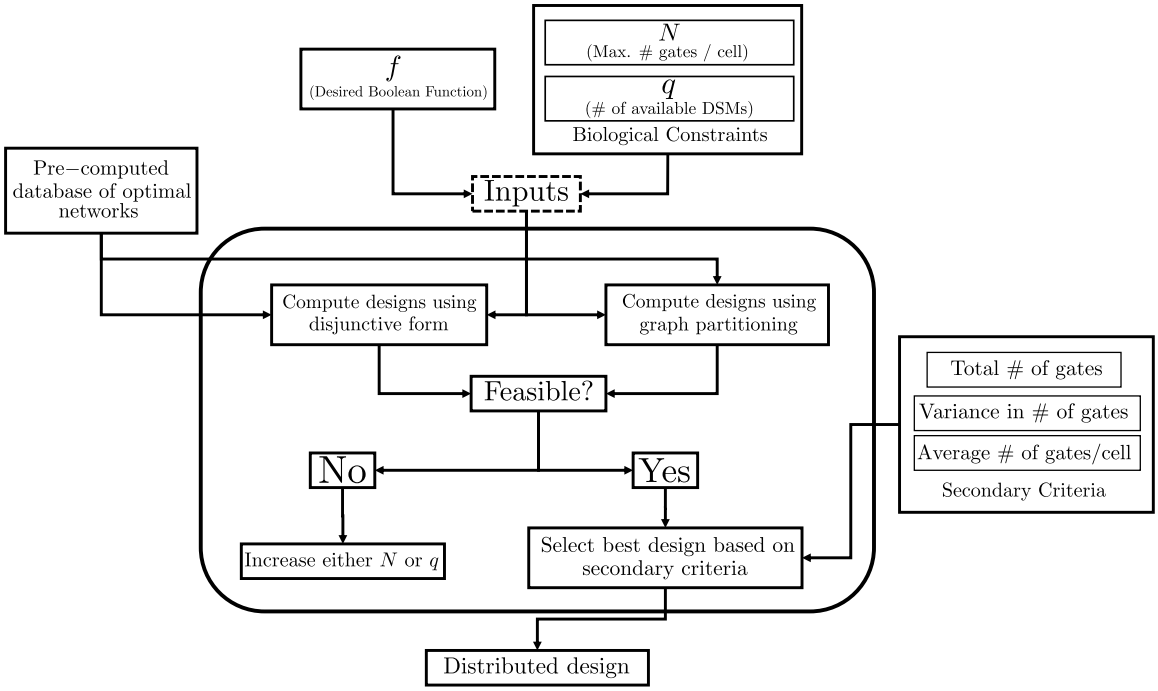


Figure 5.4: A flowchart of the optimized distributed design framework.

5.3.6 Illustrative examples and Pareto trade-offs

In this section, we show various circuit realizations generated by the developed framework for different 4-input BFs under the constraint that $N = 7$ starting with $(7, 1)$- up to $(7, 3)$-realizable circuits. Examples of $(7, 0)$-realizable circuits are omitted here because they can be looked-up directly through the database and they do not involve distributed computation.

We first start with the case of a single DSM. In Fig. 5.5(a) and (b), two $(7, 1)$-realizable circuits are shown with both circuits having six and seven product terms in their MDF. However, given the look-up table, these realizations are reduced from requiring six and seven cells in the disjunctive-form design to simply requiring two and three cells, respectively. Through combining various minterms, it was found that it is not possible to decrease the maximum NOR gates requirement. In other words, if a minterm is offending, it was not possible for us to find a combination of this minterm with the other minterms of a given BF to realize it with a smaller number of NOR2 gates. However, through these examples, we illustrated that it can be quite straightforward to reduce greatly the required number of cells through the exhaustive

search of the optimized disjunctive-form design.

The next examples show the cases in which more than one DSM is needed and how the algorithm handles offending minterms. Fig. 5.6 depicts disjunctive form realizations of a BF for which a DSM is necessary to deal with the offending minterm. In Fig. 5.6(a), Cell 5 is used to create the first DSM signal, which feeds as an input to Cell 4. Cells 1 through 4 combine their outputs through a DSM sensor that acts as an OR gate. Fig. 5.6(b) shows an example dealing with two offending minterms for which the resulting optimized DFD is (4,2)-realizable.

In Fig. 5.7, two different DSMs are used to deal with the three offending minterms. In that design, cells #2 and #3 share a similar input circuit releasing a common DSM. The remaining four non-offending product terms are combined to form two cells requiring seven NOR gates each. This (7,3)-realizable circuit is distributing the computation in two steps by first computing the DSM1 and DSM2 signals from cells 1 and 2, and then computing the operations in cells 3 through 7.

While only a few examples are shown here, the developed algorithm can generate optimized realizations for all 65536 possible 4-input BFs.

To illustrate the flexibility of the workflow, various designs are shown in Fig. 5.8 that can be generated from the DFD form depending on the criteria of selection that can be, for example, the average number of gates per cell, the total number of gates in the design or the variance in the number of gates in final design.

Also, in order to demonstrate the complementary nature of a unified algorithm that integrates the partitioning and DFD approaches, we show in Fig. 5.9 how the DFD decomposition of a circuit can yield a (7,1)-realizable design that would not be otherwise possible with a partitioning algorithm. In Fig. 5.10, the opposite scenario is shown, in which the partitioning algorithm of the full circuit provided by the exact synthesis algorithm yields a very compact (7,1)-realizable design of 2 cells using 5 and 7 gates. A comparable design in the DFD would require at least 3 cells and 2 DSMs.

A 6-output 4-input design example. We consider a circuit with two 2-bit binary numbers y_1y_0 and x_1x_0 as inputs. As an example, let us design a full adder ($c_2c_1c_0$), a

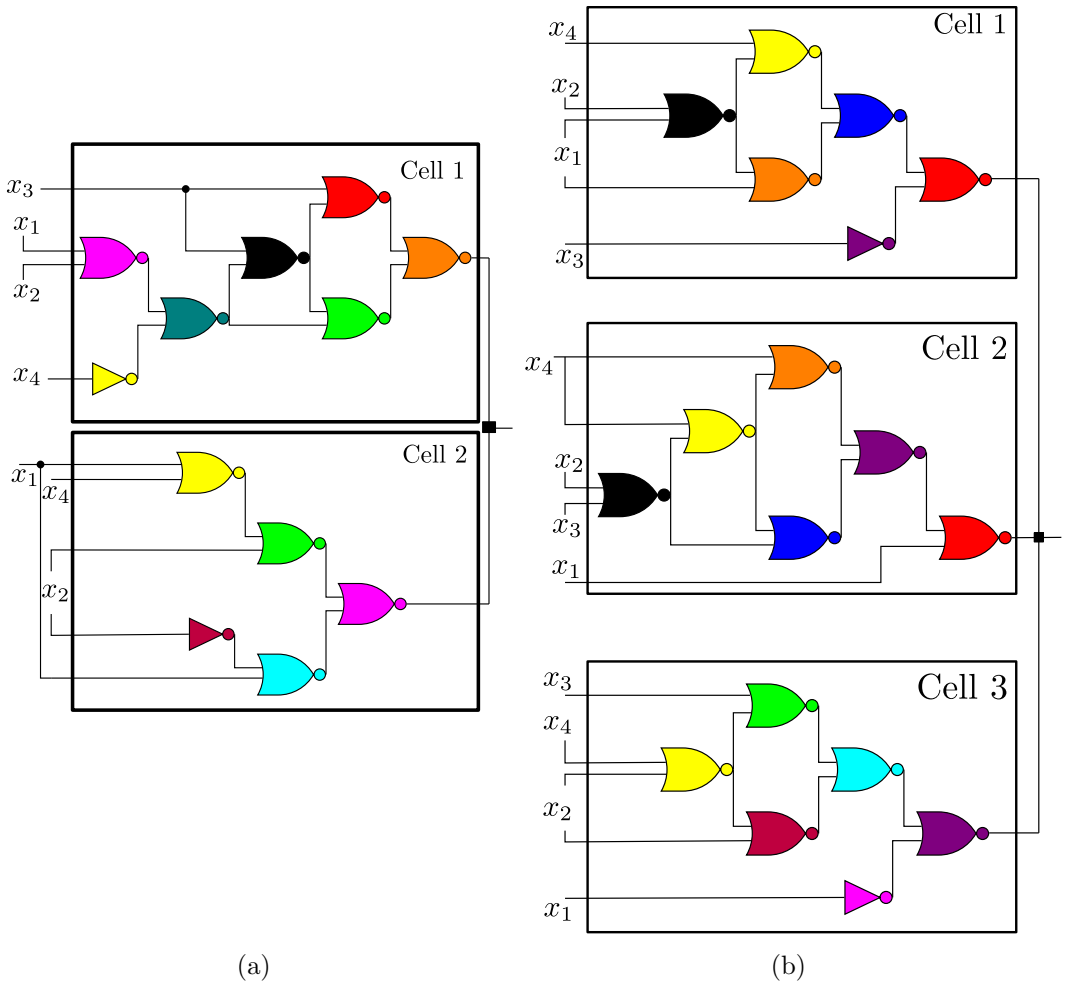


Figure 5.5: Distributed designs with only one DSM (i.e., (7,1)-realizations) using the proposed algorithm for two Boolean functions. (a) Distributed realization of the Boolean function with the hexadecimal representation (0xE99F), or equivalently $f(x) = x_1x_2x_3 + x_3x_4 + x_2x_3x_4 + x_1x_3x_4 + x_1x_2 + x_1x_2x_4$, that requires at least 13 gates to be implemented in a cell. The corresponding graph cannot be partitioned with one DSM. Nevertheless, the optimized disjunctive form provides a design of two cells with 7 and 5 gates as depicted above. This is notable because the distributed design requires less gates than the full circuit using NOR2 gates. (b) The Boolean function (0x977E) requires at least 14 gates to be implemented in a cell. The optimized design yields a design using three cells of 6 gates each. The MDF form is given by $f(x) = x_1x_2x_3x_4 + x_1x_3x_4 + x_1x_2x_4 + x_1x_2x_3 + x_1x_3x_4 + x_1x_2x_4 + x_1x_2x_3$. The symbol "■" refers to a DSM sensor.

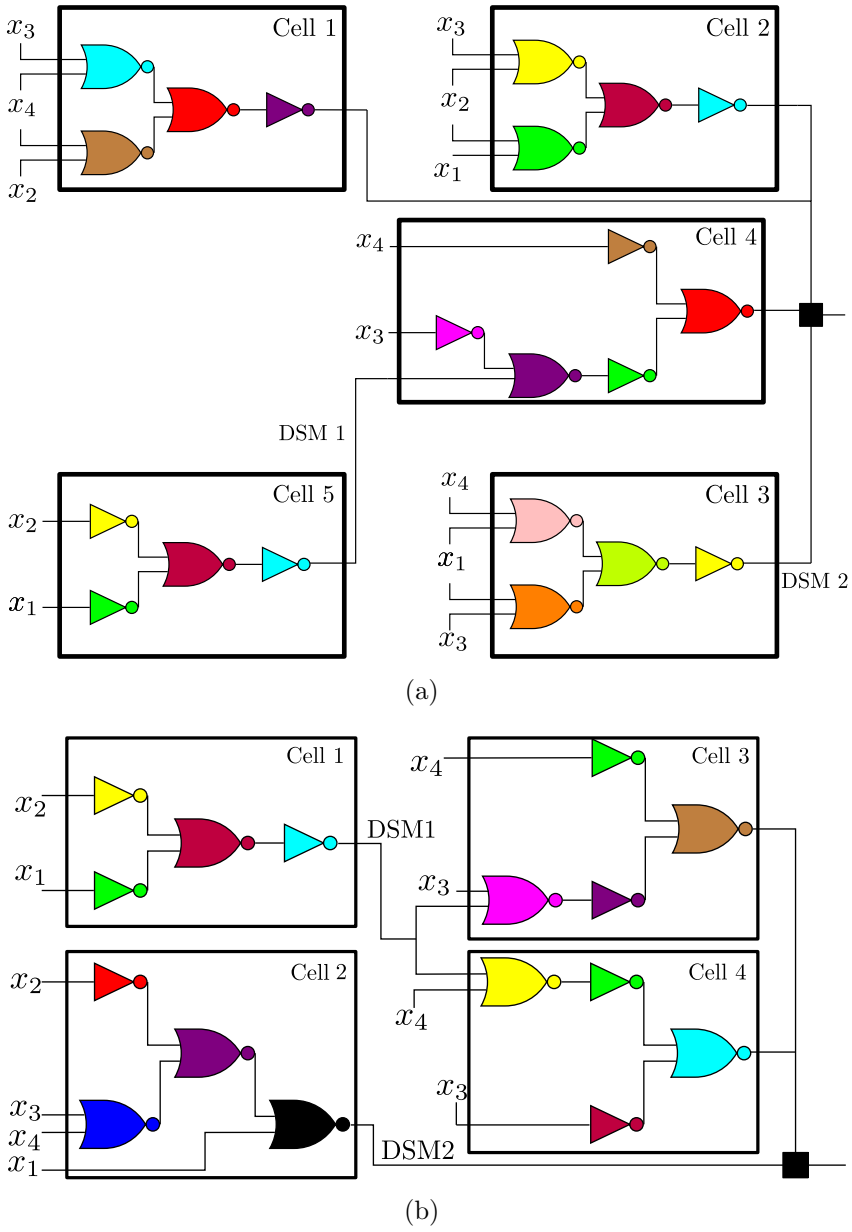


Figure 5.6: Distributed designs with two DSMs (i.e., (7,2)-realizations) using the proposed algorithm for two Boolean functions with one and two offending minterms. (a) The function (0xFEE9) requires 14 gates to be implemented in a cell. The MDF form is provided by $f(x) = x_1x_2 + x_2x_3 + x_3x_4 + x_2x_4 + x_1x_4 + x_1x_3 + x_1x_2x_3x_4$. The optimization procedure was used to reduce the number of required cells from 7 (based on the MDF form) to 5, while also ensuring that the final design is (5,2)-realizable. The symbol "■" refers to a DSM sensor. (b) The BF (0xF806) requires 11 gates to be implemented in a cell, and the resulting graph cannot be partitioned with one cut. The MDF form is provided by $f(x) = x_1x_2 + x_1x_2x_3x_4 + x_1x_2x_3x_4 + x_1x_3x_4$. The optimized design required four cells with a unique "sender" (Cell 1) that acts upon two "receivers" (Cells 3,4), making this design (4,2)-realizable.

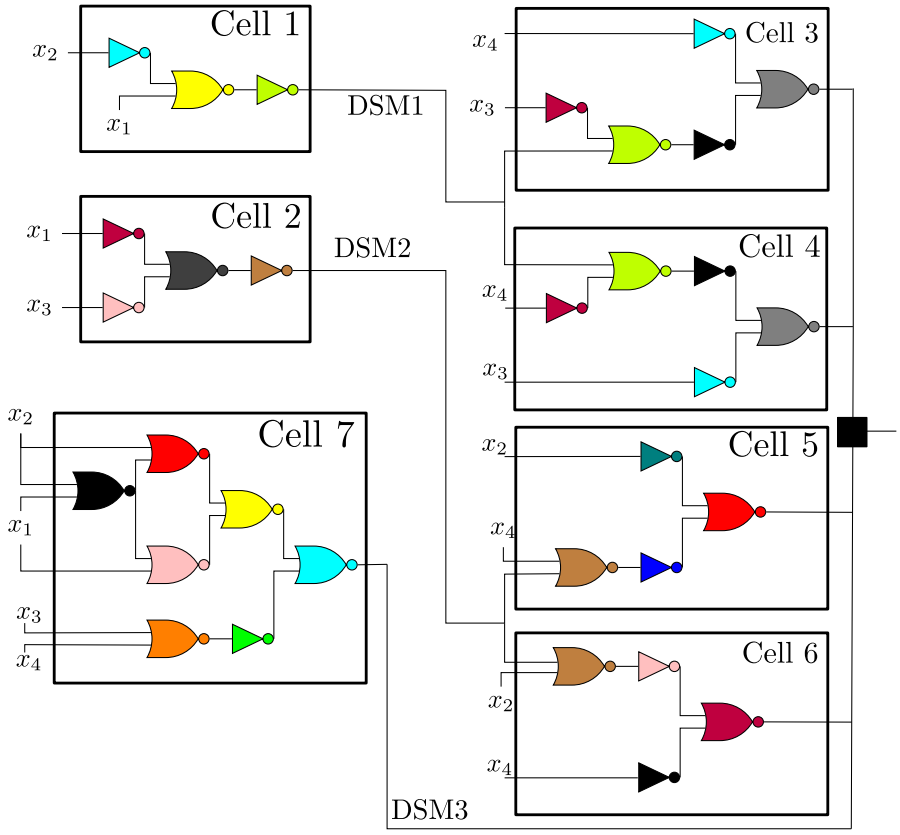


Figure 5.7: Distributed designs with three DSMs (i.e., (7,3)-realizations) using the proposed algorithm. This function (0x2196) requires at least 13 gates to be implemented in a cell. The MDF is provided by $f(x) = x_1x_2x_3x_4 + x_1x_2x_3x_4 + x_1x_2x_3x_4 + x_1x_2x_3x_4 + x_1x_2x_3x_4 + x_1x_2x_3x_4$. The symbol "■" refers to a DSM sensor.

subtractor $(d_1 d_0)$ and a comparator. The outputs are defined as: $(c_2 c_1 c_0) = (y_1 y_0) + (x_1 x_0)$, $(d_1 d_0) = |(x_1 x_0) - (y_1 y_0)|$, where addition and subtraction here refer to the standard operations on the field of real numbers. The comparators are defined as follows: $e = 1$ iff $(y_1 y_0) = (x_1 x_0)$ (and $e = 0$ otherwise), and $g = 1$ iff $(y_1 y_0) > (x_1 x_0)$ (and $g = 0$ otherwise). A distributed design that shares minterms between the different BFs is shown in Fig. 5.11.

5.4 Discussion

Despite impressive progress in the technology of rational construction of synthetic genetic networks, the implementation of large logical decision-making circuits has been problematic, due to repressor toxicity and the lack of large sets of mutually-orthogonal repressors. Indeed, a typical circuit contains no more than roughly seven repressor-based gates per cell. In comparison, the implementation of the 4-input AND gate requires at least 9 NOR2 gates, and up to 14 NOR2 gates for an arbitrary 4-input BF. Hence only a tiny fraction of 4-input Boolean functions is within the reach of the current technology. One way to overcome these limitations is to distribute processing into multiple cell types, which communicate among themselves using quorum sensing or other diffusible small molecules (DSMs). Here, we developed a systematic framework to implement such a distributed approach to synthesize distributed realizations, for any Boolean function of several inputs (such as a combination of chemical and physical signals), and under constraints on the maximal number of gates per cell and a maximal number of orthogonal DSMs. We have shown that cell-to-cell communication increases the number of implementable Boolean functions significantly. For example, with at most seven gates per cell and four inputs, a single DSM increases the number of realizable circuits by at least 7.58-fold compared to centralized computation, and with two DSMs it is possible to implement almost all four-input Boolean functions.

Our approach applies equally well to any number of inputs larger than four. However, computational challenges make it very hard to obtain exact numbers when the

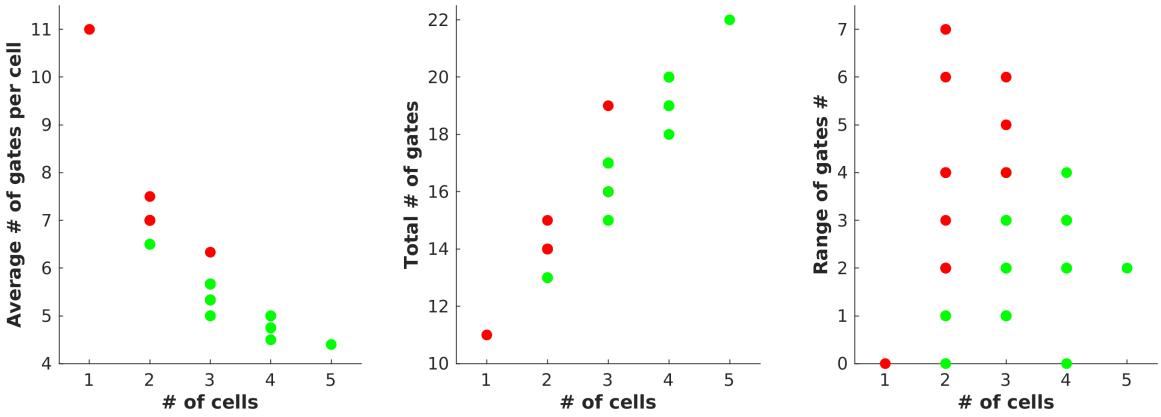


Figure 5.8: Various possible circuit designs using the exact synthesis library are shown for the BF (0xA7D4). While the focus here was on minimizing the total number of gates, other optimization criteria can be readily implemented. Green bullet points indicate designs that are realizable within the constraint of 7 NOR2 gates per cell, while red bullet points indicates designs which are unrealizable.

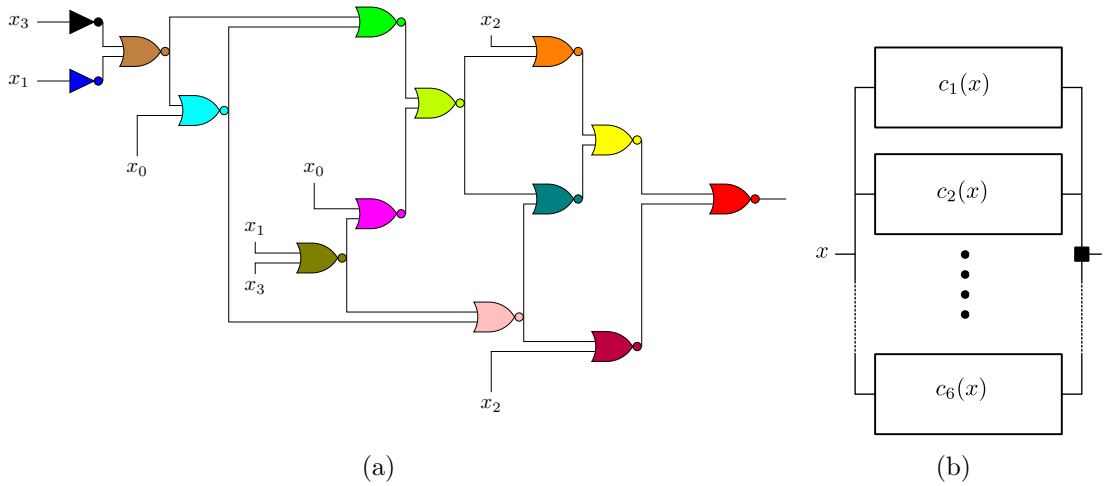


Figure 5.9: Depicted here is an illustrative example for which the DFD is a lot more economical in terms of required number of DSMs compared to graph partitioning applied to the full design in the database. (a) The exact synthesis design representing (0x1668) is shown where it requires at least 14 NOR2 gates. There exists no partitioning using 2 DSMs that can make this representation (7,2)-realizable. (b) The proposed disjunctive design re-synthesizes the logic from the bottom up and it provides a (7,1)-realization with six cells each containing 7 gates. The BFs are $c_1(x) = x_1x_2x_3x_4, c_2(x) = x_1x_2x_3x_4, c_3(x) = x_1x_2x_3x_4, c_4(x) = x_1x_2x_3x_4, c_5(x) = x_1x_2x_3x_4, c_6(x) = x_1x_2x_3x_4$. The symbol "■" refers to a DSM sensor.

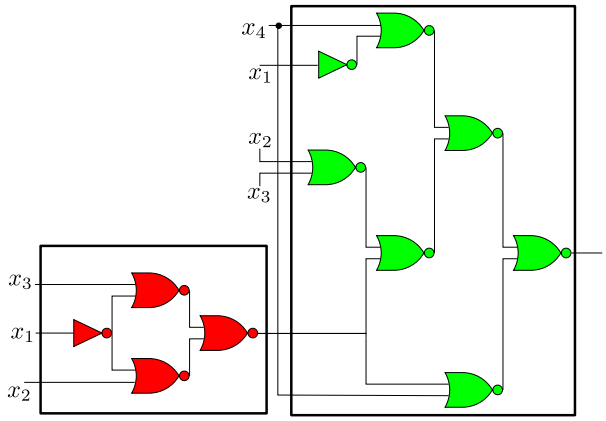


Figure 5.10: Depicted here is an illustrative example for which the design using the partitioning algorithm is more compact than the most compact DFD found. The exact synthesis design for (0x0016) is shown. Highlighted by the red and green colors is the partitioning of this circuit into two cells requiring only 5 and 7 gates and 1 DSM. The alternative design (not shown) given by the disjunctive form approach needs a higher requirement of 3 cells and 2 DSMs.

number of inputs is large; thus, as the number of inputs increase, one must settle for suboptimal sizes and random sampling of Boolean functions, as opposed to the exhaustive study that is feasible for four inputs.

Future research will study heuristic algorithms for sampling of suboptimal designs using the approach discussed here, and the comparison of these algorithms with existing ones –for the case of no DSMs– such as ABC or Espresso (which result in considerably suboptimal designs). Another future direction is the adaptation of our current framework to be conducted on massively parallel processing devices. By expanding the ability of these algorithms to test hundredfold or thousandfold more circuits in the same time span, one would expect even an exhaustive exact synthesis algorithm to be able to calculate optimal 5- and 6-input circuits within a reasonable time frame, while also improving the computing of larger sub-optimal designs. We view this work as a first step in a rational design theory for distributed cellular computation.

The database as well as well as the toolbox that we created to generate the various designs in this work are made available at `https://github.com/sontaglab/DBC/`.

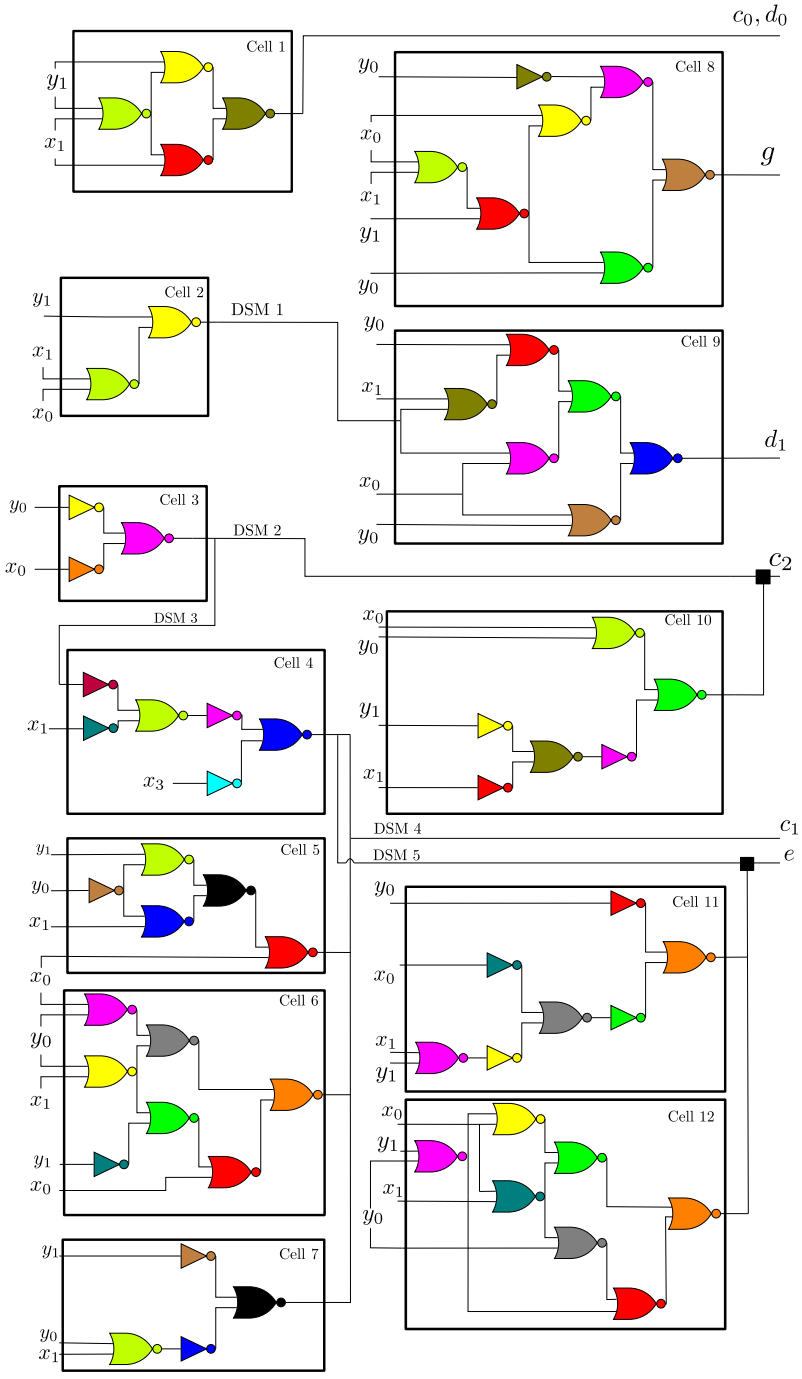


Figure 5.11: A 6-output 4-input circuit that uses 5 DSMs. The circuit implements a 2-bit adder, a 2-bit subtractor and a comparator. The outputs are defined as: $(c_2c_1c_0) = (y_1y_0) + (x_1x_0)$, $(d_1d_0) = |(x_1x_0) - (y_1y_0)|$, where addition and subtraction here refer to the standard operations on the field of real numbers. The comparators are defined as follows: $e = 1$ iff $(y_1y_0) = (x_1x_0)$, and $g = 1$ iff $(y_1y_0) > (x_1x_0)$. The symbol "■" refers to a DSM sensor.

5.5 Application example: 3-bit adder

In this section, the implementation of a 3-bit adder is explored in details. First, an implementation in cascade is explored in which the 1^{st}, 2^{nd}, and 3^{rd} bits are calculated in succession. This calculation is usually broken down into a carry term and a sum term, the later being the result of the addition. Because in a cascade setting, the previous result is needed to calculate the next one, the ideas of carry look-ahead which calculates the carry terms in parallel is also explored. Finally, we look at how one would implement the result of the 1^{st} and 2^{nd} bits directly. This exercise is a demonstration of how this workflow can be applied in practical examples.

5.5.1 Implementation of self-contained adding units in cascade

A full adding unit can be implemented in the following way:

$$S_i = A_i B_i' C_{i-1}' + A_i' B_i C_{i-1}' + A_i B_i C_{i-1} + A_i' B_i' C_{i-1} \tag{5.3}$$

$$C_i = A_i B_i' C_{i-1} + A_i' B_i C_{i-1} + A_i B_i \tag{5.4}$$

with S_i being the sum and C_i the carried value.

Using the distributed computation workflow, we found that S_i is (9,0) and C_i is (6,0). When the limit of gates per cell is 7, C_i can be readily implemented as shown in Fig. 5.15. The implementation of S_i can be done with two cells in the arrangement shown in Fig. 5.14.

Then one can cascade these full-adder units (Fig. 5.12) to make a 3-bit adder (Fig. 5.13).

5.5.2 Implementing S_0 and C_0

In the absence of a "feeding" C value, the circuits simplify to:

$$S_0 = A_0 B_0' + A_0' B_0 \tag{5.5}$$

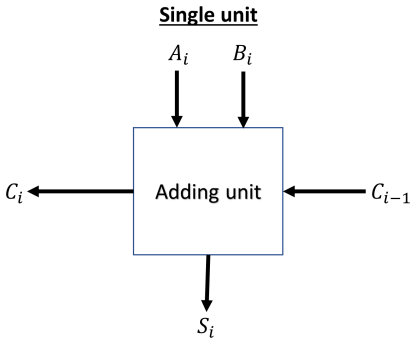


Figure 5.12: A single adding unit.

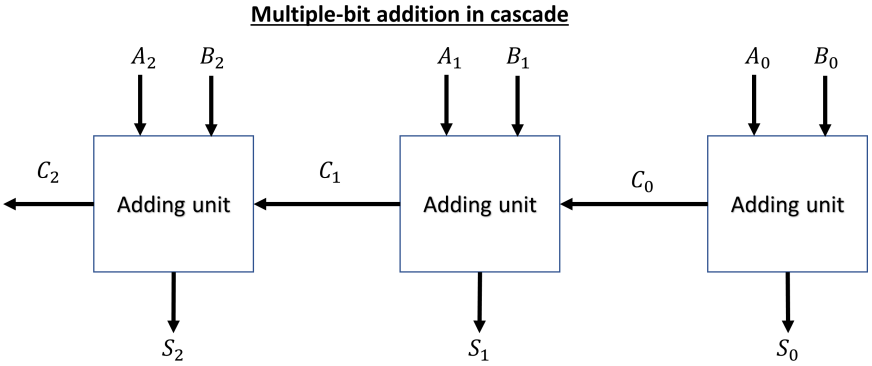
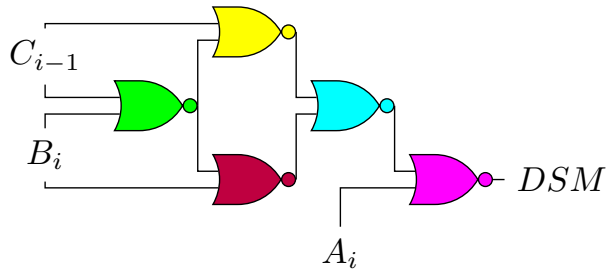


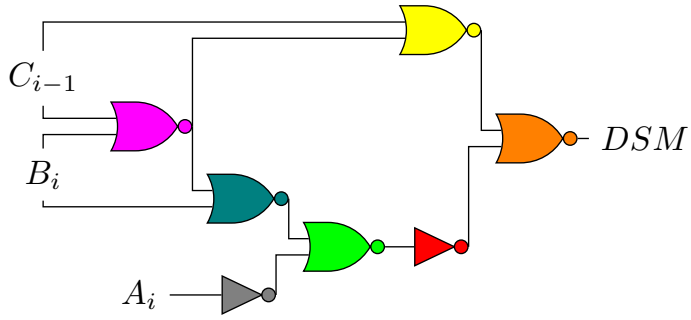
Figure 5.13: Three adding units in cascade.

$$C_0 = A_0 B_0 \tag{5.6}$$

The most right/conservative bit can then be implemented using the following reduced circuits. The circuits for S_0 and C_0 are provided by Fig. 5.16 and Fig. 5.17, respectively.



(a)



(b)

Figure 5.14: Implementation of S_i

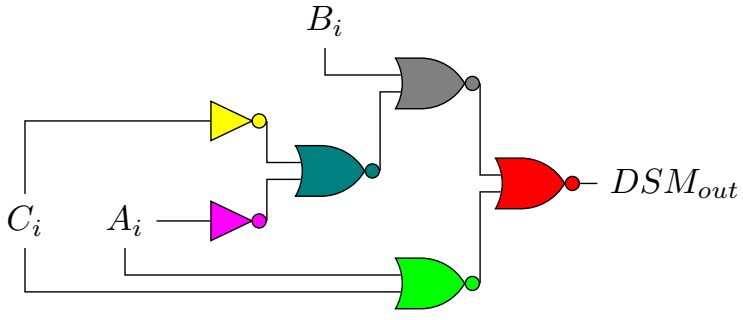


Figure 5.15: Implementation of C_i.

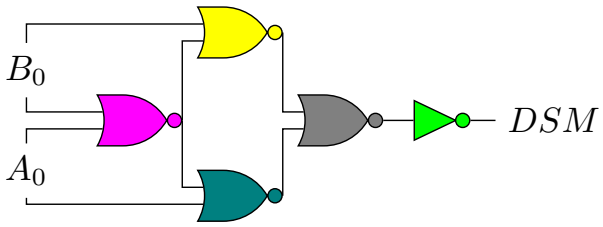


Figure 5.16: Implementation of S_0.

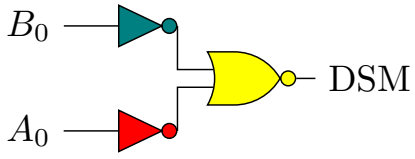


Figure 5.17: Implementation of C_0.

5.5.3 Delay reduction: calculating carry values in parallel

The sequential implementation of the previous adder can introduce significant delay. By introducing the variables:

$$P_i = A_i + B_i \tag{5.7}$$

$$G_i = A_i B_i. \tag{5.8}$$

The carry terms can be redefined such that:

$$C_0 = G_0 \tag{5.9}$$

$$C_1 = G_1 + P_1 G_0 \tag{5.10}$$

$$C_2 = G_2 + P_2 G_1 + P_2 P_1 G_0. \tag{5.11}$$

C_0 is shown in the previous section.

C_1 can be decomposed into the following DNF form:

$$C_1 = A_0 A_1 B_0 + A_0 B_0 B_1 + A_1 B_1. \tag{5.12}$$

The C_1 implementation is shown in Fig. 5.18.

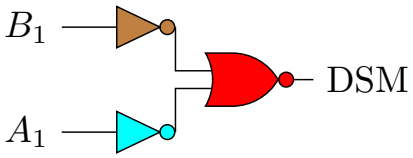
$$C_2 = A_0 A_1 A_2 B_0 + A_0 A_1 B_0 B_2 + A_0 A_2 B_0 B_1 + A_0 B_0 B_1 B_2 + A_1 A_2 B_1 + A_1 B_1 B_2 + A_2 B_2. \tag{5.13}$$

The first 4 terms of C_2 can be computed using the generic 4-input AND gates. This is 9 gates arrangement that can be partitioned into 2 circuits using DSMs.

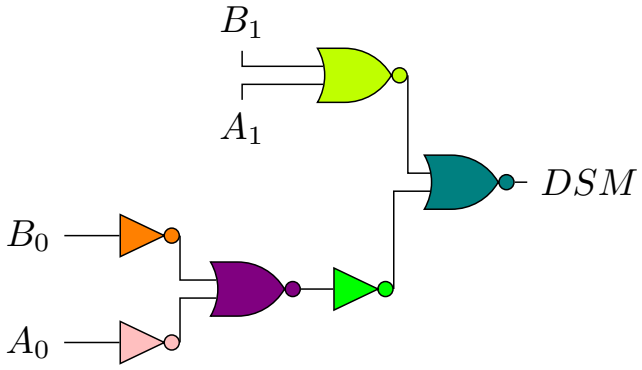
Note that the first two terms $A_0 A_1 A_2 B_0$ and $A_0 A_1 B_0 B_2$ have the redundancy $A_0 A_1 B_0$. The next two terms $A_0 A_2 B_0 B_1$ and $A_0 B_0 B_1 B_2$ has the redundancy $A_0 B_0 B_1$.

In practice, these 4 minterms translate into 2 “sending” cells and 4 “receiving” cells.

In addition, the last three terms $A_1 A_2 B_1 + A_1 B_1 B_2 + A_2 B_2$ are represented by the



(a)



(b)

Figure 5.18: Implementation of C_1.

same circuit as the one for C_1 with ordering (A_1, B_1, A_2, B_2).

In total, the computation of C_2 requires 8 types of cell that are shown in Fig. 5.19.

5.5.4 Direct computation of the 2^{nd} bit S_1

S_1 can be taken as 4-input function. Using a truth table of 2-bit sum, the binary representation of this circuit is "0011011011001001" which corresponds to a (11,0) circuit. This distributed implementation is shown in Fig. 5.20.

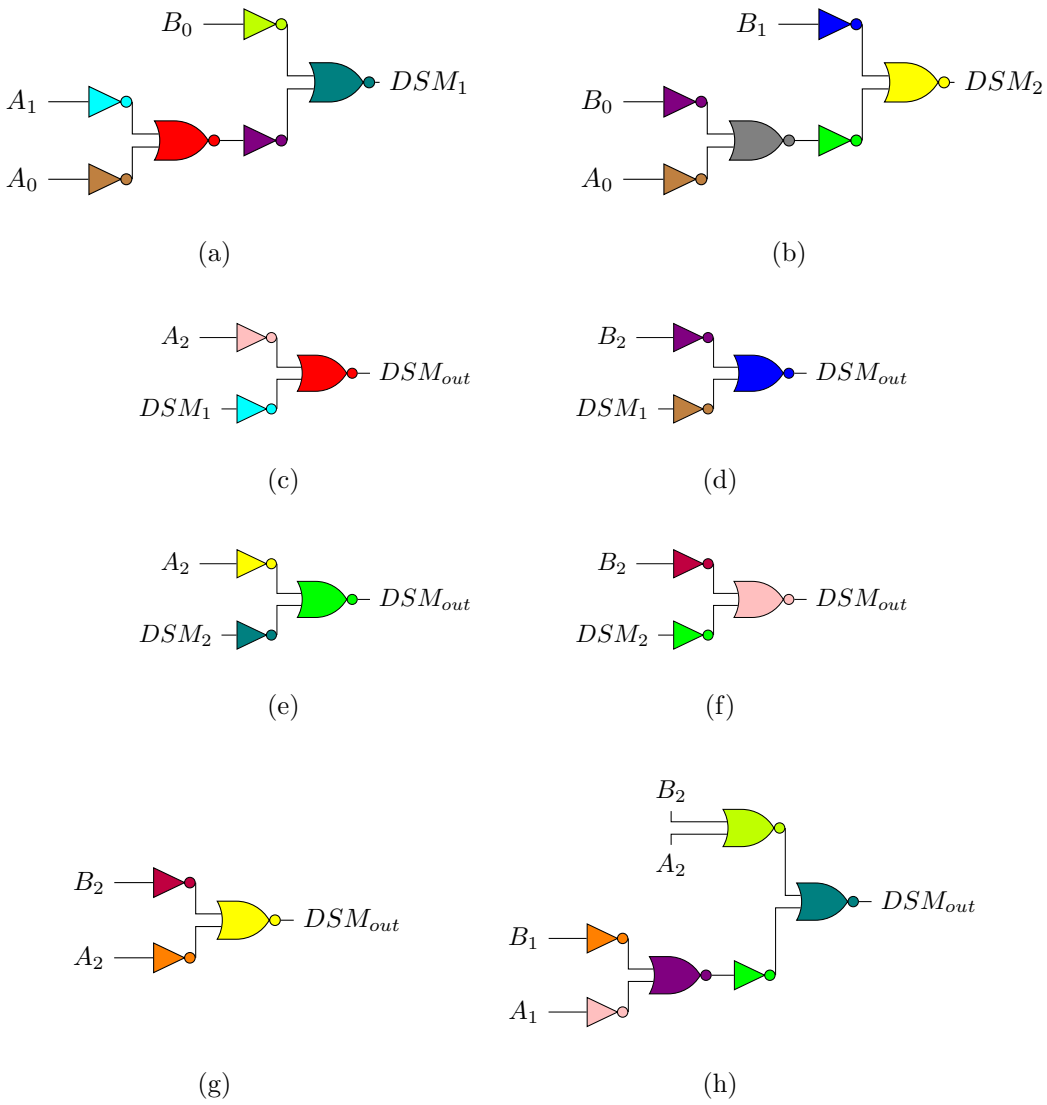


Figure 5.19: Implementation of the C_2 circuit.

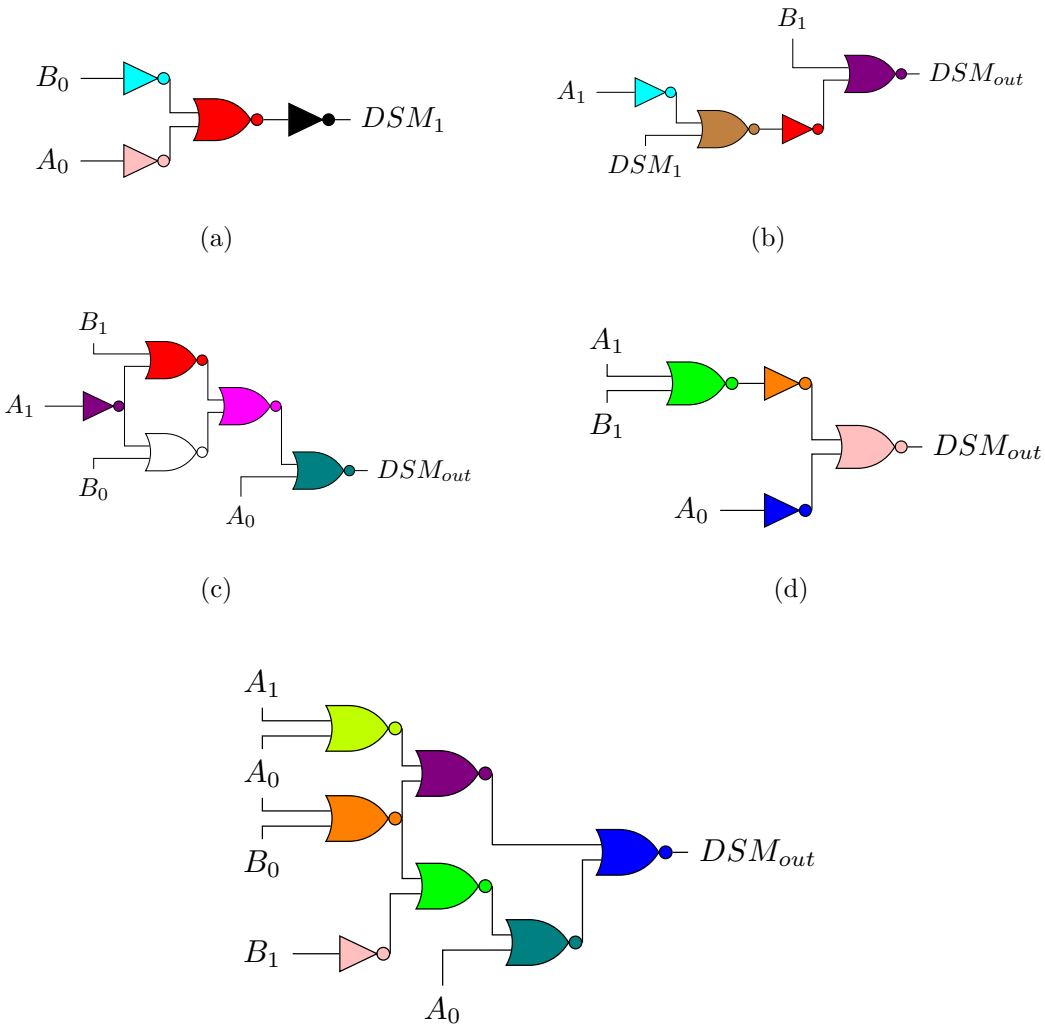


Figure 5.20: Implementation of the S_1 circuit.

Chapter 6

Optimization of heuristic logic synthesis by iteratively reducing circuit substructures using a database of optimal implementations

6.1 Motivation of this work

To extend the idea of distributed computation to Boolean functions with a number of inputs greater than 4, we introduce an algorithm that takes the best of both the heuristic and optimal multi-level synthesis and combine it into a hybrid framework that yields more efficient circuits. The optimal circuit implementations are based on the thesis work of Ernst [168] and are used to improve the results provided by *ABC* and making the final designs significantly more efficient for medium size circuits. The importance of reducing circuit size is of particular importance, but not exclusively, to the area of synthetic biology where a few less gates can be the difference between having a biologically implementable circuit layout or not [113, 194].

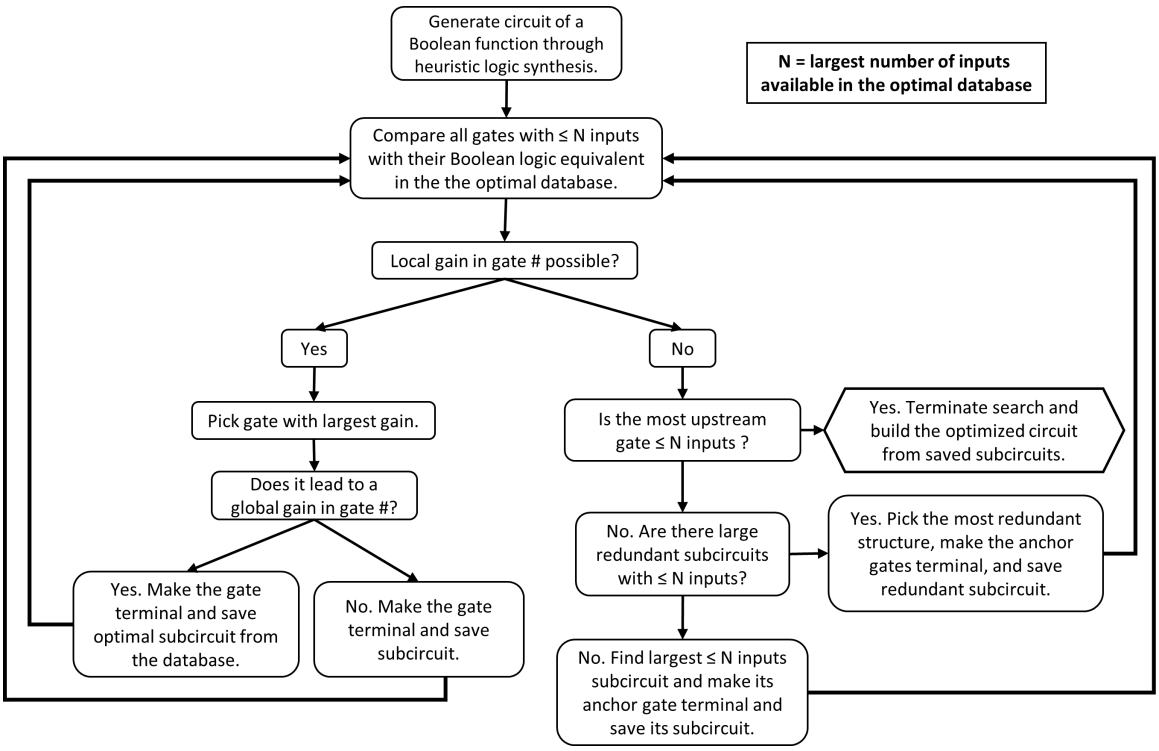


Figure 6.1: A flowchart of the hybrid logic minimizer algorithm.

6.2 Methodology

6.2.1 Database of optimal synthesis implementations

In the context of this work, the definition of "optimal" pertains to the optimality of a circuit (in the sense that it has a minimal number of unique gates among all the circuits implementing a given function). The Branch-and Bound Exact Synthesis System (*BESS*) algorithm was used to develop a database of optimal NAND with fan-in of 2 (NAND2) circuits for all 2-, 3-, and 4-input Boolean functions [168, 113]. This database was used to readily generate the corresponding optimal 2-input NOR circuits (NOR2) database using the appropriate circuit transformations. The limitation in building a very large database stems not only from the lengthy exhaustive search that even an individual 5-input Boolean function requires but also from the fact that the number of database entries necessary grows too rapidly to be practical with larger circuits. This is most easily seen by observing that there are 68 P-equivalent 3-input functions, 3,904 4-input functions, and 37,329,264 5-input func-

tions (two Boolean functions are P-equivalent if one function can be transformed into the other by permuting the input variables). In addition to the issue of calculating the individual implementations of Boolean functions, a larger database also causes issues with storage and access. For these reasons, the investigation was focused on using an optimal implementation database for only up to 4-input Boolean functions. Let us note that there can be multiple optimal implementations of the same Boolean logic. The focus of this work was limited to the first implementation that is found through the *BESS* algorithm.

6.2.2 Heuristic logic synthesis

To get the circuit implementations for Boolean functions with more than 4 inputs, the well-established *ABC* algorithm was used [35]. The commands used in *ABC* in order are: *strash*, *rewrite*, *refactor*, and *balance*. Our framework uses *MATLAB* as an interface to create the input files, launch the *ABC* simulations through the command line, to parse the results from the output files, and to convert from an AND-Inverter Graph (AIG) to a NOR-Inverter Graph (NIG).

6.2.3 Hybrid logic minimizer algorithm

The main idea of this algorithm is to look for subcircuits that can be simplified using the optimal synthesis database. A schematic of the algorithm is shown in Fig. 6.1.

Given that the optimal database does not contain entries for circuits containing more than 4 inputs, the focus of this algorithm is limited to substructures of 4 input or less. Even with this limited set of substructures, analyzing the possibilities exhaustively would still be problematic. In addition to potentially having to identify all the possible substructures of 4 inputs or less, the order in which these substructures are to be replaced by their optimal counterparts can lead to different outcomes in the observed overall gate reduction. The algorithm presented here does not aim to reduce the number of overall unique gates optimally but rather aims to reduce the search space and finding a "good" solution rapidly. The algorithm is subdivided into three

steps that are checked successively at each iteration (in an *if*, *else if*, *else* pattern):

Substitute an eligible substructure with its optimal counterpart

Taking the circuit to be analyzed and thinking of gates in terms of nodes, any parent node that has children nodes totaling more than 4 inputs does not qualify to be optimized at this step. Evidently, any parent node containing a child node that does not qualify also does not qualify. By doing this, the algorithm only has to search substructures that are close to the terminal nodes of the circuit, limiting greatly the complexity of the search. The circuit of each eligible node formed by the node and all of its descendants is then compared with the NOR2 optimal database and the difference of gate numbers is recorded. At the end of this step, the algorithm replaces the circuit with the largest recorded difference in gate numbers by its optimal counterpart. The overall circuit to be considered in subsequent steps has the root of the replaced circuit replaced by a terminal node as the algorithm no longer needs to optimize this part of the circuit. At the next iteration of the search, the size of the considered circuit or netlist is thus reduced.

The optimized circuit is tested at the end of this step by replacing the altered terminal nodes by the corresponding subcircuits to rebuild the entire logic circuit. This 'global' implementation of the Boolean logic is used to check whether the local improvement led to a global improvement in terms of reducing the number of unique gates. If it fails to do so, the original substructure that was replaced is saved as subcircuit instead of the one provided by the database. This ensures that that the circuit unique gate count cannot be made worse by this step.

Substitute all occurrences of a redundant optimal substructure

When no direct improvement can be found, it also indicates that eligible substructures are found to be optimal with respect to the database. In order to continue shrinking the search space, the algorithm attempts to find substructures to substitute (in other words, replacing the root node of this substructure by a terminal node in the considered circuit). The choice of substructures to be substituted can be quite large.

The main issue with replacing a substructure at random is that the root location where the replacement takes place introduces a new input to the overall circuit. It is apparent that having too many unique inputs to a circuit reduces the ability of the algorithm to execute Step 1. Thus, our implementation looks for the 3-input or 4-input substructure that is most redundant in the circuit in terms of occurrences. By doing this, the search space is shrank, new eligible nodes are created for Step 1, the replaced substructure is optimal, and the same new input is introduced at all the replaced root nodes.

Substitute a substructure that maximizes the reduction in the number of unique inputs in the netlist

At this step, there is neither a readily available substructure to optimize nor are there eligible redundant substructures. Let us note that by substituting a substructure that may shrink the number of unique inputs in the considered circuit, it is also increasing the chances that Step 1 finds a substructure to be optimized at the next iteration.

In our implementation, a list of unique inputs is made and the number of times an input appears in the circuit is recorded for each of the unique inputs. This information is then used in the algorithm to remove a 3-input or 4-input substructure that has the lowest mean occurrence across its inputs. In order words, the mean of the input frequency information is used to rank the circuits and remove the circuit with the lowest mean occurrence value. It is no guarantee of reducing the number of unique inputs but it was found to be robust in attempting to remove less frequent inputs. Similarly to Step 2, it also removes an optimal substructure to shrink the overall search space annd tries to increase the number of eligible substructures for Step 1.

6.2.4 Overall execution

Steps 1 through 3 are run iteratively until the considered circuit can no longer be shrunk. The complete circuit is then reconstituted from its substituted subcircuits. The details of the implementation are included with the computational package.

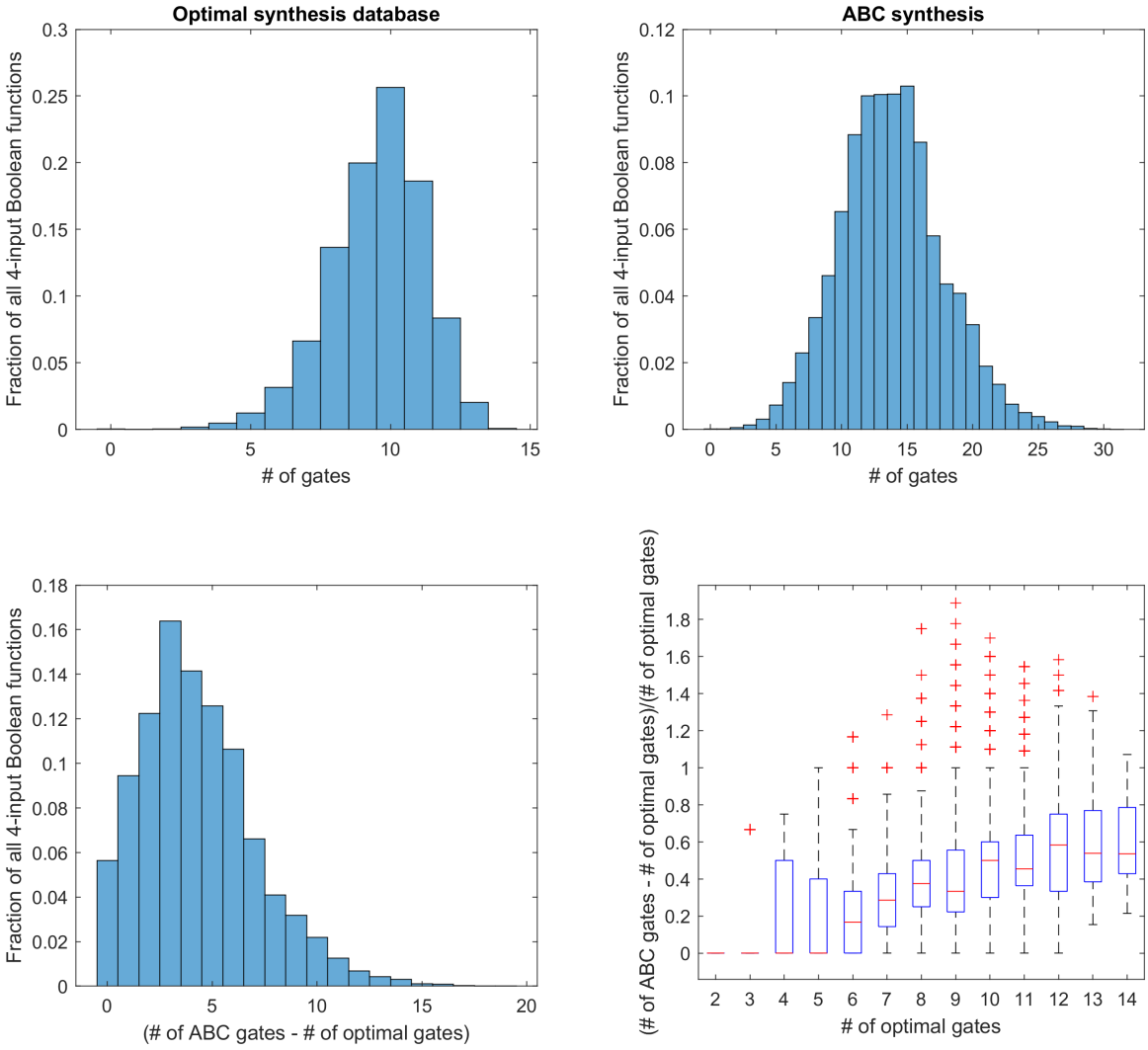


Figure 6.2: Comparison in the number of gates for NOR2 circuits generated from ABC and those generated through optimal synthesis for all 4-input Boolean functions. (a) and (b) show the distributions in the number of gates for the optimal synthesis database and the ABC synthesis, respectively. (c) shows the distribution in the difference in gate numbers between the two results. In (d), a box plot is drawn of the deviation from optimal normalized by the number of optimal gates as a function of the number of optimal gates.

6.3 Results and Discussion

Before considering the performance of the proposed algorithm, *ABC* was used to generate all the possible 65,536 4-input Boolean functions and the results were compared with the entries of the developed optimal database of NOR2 circuits. These results are summarized in Fig. 6.2. On average, *ABC* produces circuits that require 4.310

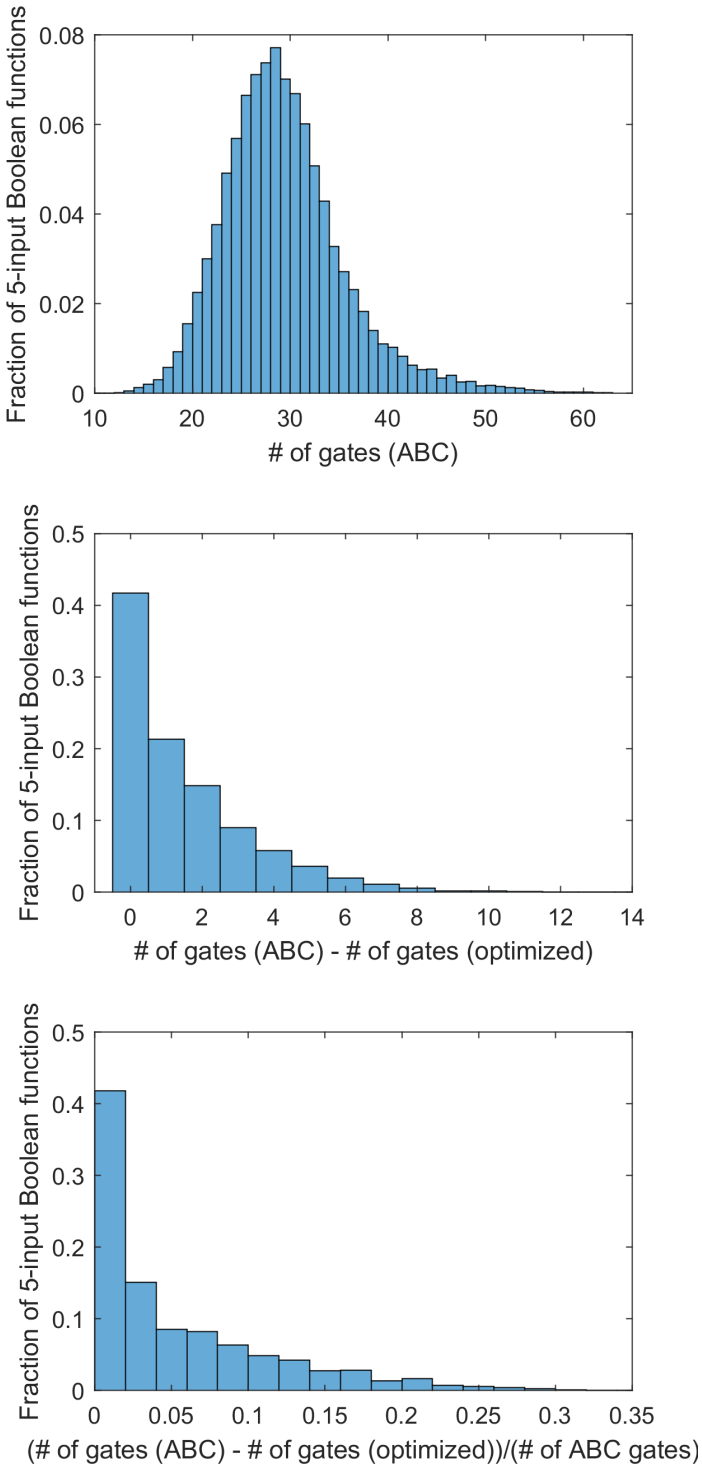


Figure 6.3: 20,000 5-input NOR2 circuits are generated at random from ABC and optimized using the described workflow. (a) shows the distribution of the number of gates in the ABC circuits. (b) is a histogram of the improvements made by the proposed algorithm in terms of number of gates. (c) is a histogram of the fractional improvement made by the proposed algorithm.

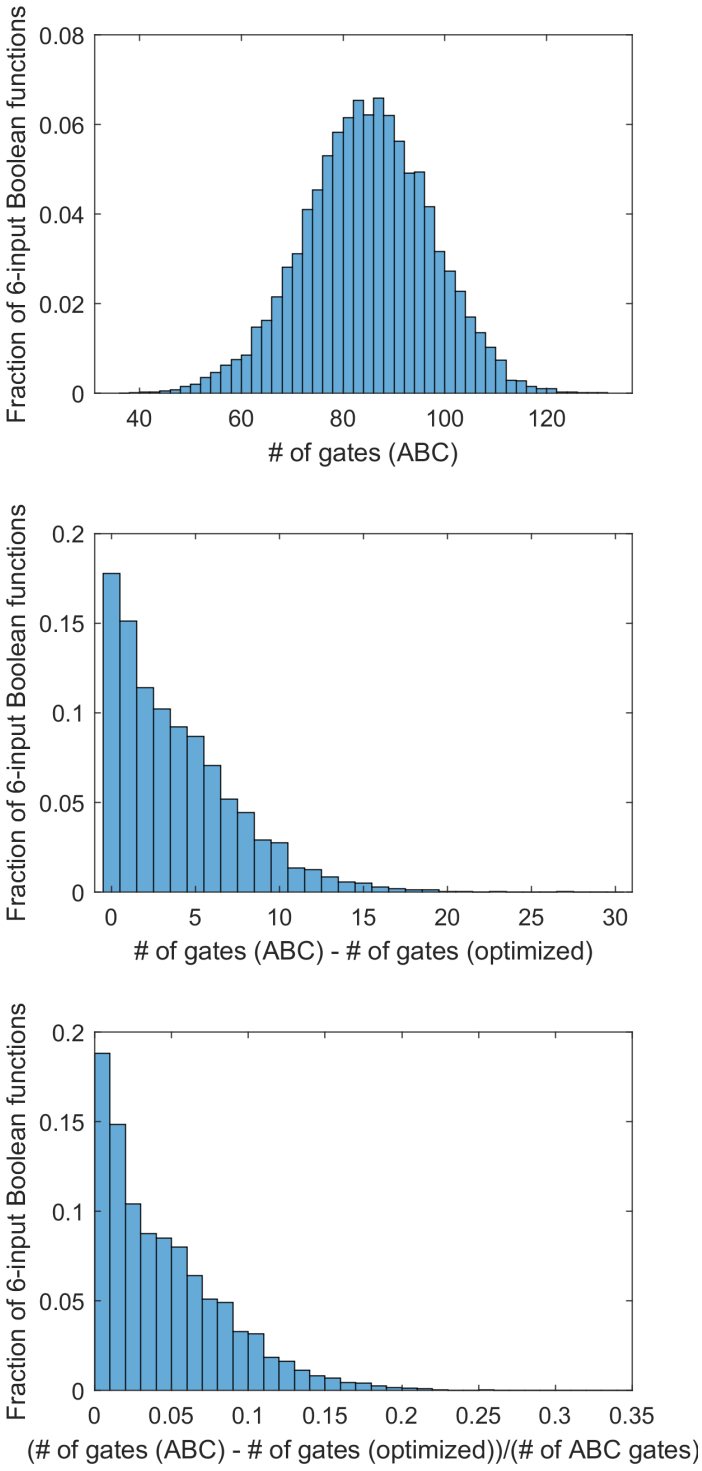


Figure 6.4: 12,000 6-input NOR2 circuits are generated at random from ABC and optimized using the described workflow. (a) shows the distribution of the number of gates in the ABC circuits. (b) is a histogram of the improvements made by the proposed algorithm in terms of number of gates. (c) is a histogram of the fractional improvement made by the proposed algorithm.

additional gates and only finds the optimal solution 5.630% of the time. The mean number of gates are 9.514 and 13.824 gates for the circuits derived from the optimal synthesis and *ABC*, respectively. From the box plot in Fig 6.2(d), the trend is clear that larger optimal circuits also lead to a larger deviation from optimality for the *ABC* implementations, primarily due to increased complexity. This gives an idea of what is the theoretical maximum gain in terms of gate number.

The algorithm is then tested on 20,000 random 5-input Boolean functions. The average number of gates in a 5-input circuit generated by ABC is estimated to be 28.89 gates. The algorithm yields on average a gain of 1.46 gates. When individually normalizing the gain by the number of gates obtained by ABC, the gain was on average 5.16%. The algorithm did not improve the underlying circuit in 41.68% of cases. Looking only at the improved circuits, the improvement was on average 2.51 gates for an average normalized improvement of 8.85%. These results are also reported as histograms in Fig. 6.3.

Similarly, the results of the algorithm on 12,000 random 6-input Boolean functions are shown in Fig. 6.4. The average number of gates for 6-input Boolean functions was found to be about 84.24 gates in *ABC*. The average gain from the algorithm is 3.87 gates or 4.54%. The algorithm fails to improve the *ABC* results in only 17.78% of cases. Looking only at the improved circuits, the gains were 4.71 gates or 5.52%.

These improvements of 5.16% for 5-input and 4.54% for 6-input Boolean functions are significant given the maturity of the field of logic synthesis. What makes this finding interesting is that the a 4-input database requires little memory storage and can be rapidly accessed. The overall algorithm takes a few seconds for 5-input functions to about a minute for 6-input functions on a single-thread of an *AMD Ryzen 9 3900X* processor. We believe that there is at least 1 or 2 order of magnitude in speed improvement possible by removing redundant calculations in the present implementation. In addition, there are many independent calculations, which should make it easy to compute in parallel, thus leading to further reduction in computational time. This also shows that even a limited 4-input database has the potential to provide gate savings for much larger circuits and thus could potentially be a valuable

addition to standard logic synthesis tools.

The 4-input results shown in Fig. 6.2 also suggest that this is far from the maximum possible improvement. From the results that were obtained, it is clear that oftentimes a possible local gain doesn't lead to an overall gain, which explains why there is such a high percentage of 5-input functions that could not be improved as the algorithm currently stands. The primary challenge resides in the fact that optimal circuits oftentimes have gates with large fan-out values (the number of gate inputs driven by the output of this logical gate) to contribute to an overall saving in the number of unique gates. In a typical *ABC* output, there remain a fair amount of redundancies in the calculated circuits. By locally replacing a subcircuit to make it more efficient, one can actually make things less efficient elsewhere due to the fan-out problem.

In future iterations of this work, we will explore ways to overcome this challenge of having gates with fan-outs greater than 1. When searching for subcircuits to replace with those of the database, one can factor in this fan-out number into the algorithm to add a global perspective to the local search. Also, an optimal implementation does not necessarily mean that it is the one that will lead to the greatest global gain in gate number for the circuit. Thus, an interesting direction is to look for near-optimal subcircuits that contain redundant logic with respect to the whole circuit. While in this work we only looked at one implementation per Boolean function, the *BESS* algorithm is able to provide all the alternative solutions given desired criteria. The promising results shown here also show that there is a real value to expanding the database of functions with 5 inputs or more. Having 5-input optimal circuits would lead to fare more efficient simplifications but also lead to having a database that would be extremely large in size. Given that 4-input optimal circuits can nowadays be computed with relative ease, 5-input optimal implementations are within reach but it will also be unlikely that one can go much beyond that while maintaining this optimal criterion. Finally, the algorithm here looks at the extremities first and moves towards the "root" of the circuit. A more thorough search of substructure would most likely lead to better results, so there are promising directions with respect to

the proposed algorithm as well.

Bibliography

[1] E. A. Ernst, *Optimal Combinational Multi-level Logic Synthesis*. PhD thesis, University of Michigan, 2009.

[2] J. Wu and D. J. Waxman, "Immunogenic chemotherapy: dose and schedule dependence and combination with immunotherapy," *Cancer Letters*, vol. 419, pp. 210–221, 2018.

[3] J. Wu and D. J. Waxman, "Metronomic cyclophosphamide schedule-dependence of innate immune cell recruitment and tumor regression in an implanted glioma model," *Cancer Letters*, vol. 353, no. 2, pp. 272–280, 2014.

[4] D. E. Reusser and E. Zehe, "Inferring model structural deficits by analyzing temporal dynamics of model performance and parameter sensitivity," *Water Resources Research*, vol. 47, no. 7, 2011.

[5] "Primary immune response." `https://en.wikipedia.org/wiki/Immune_system`. Accessed: 2020-12-08.

[6] "CAR T-cell therapy." `https://www.cancer.gov/publications/dictionaries/cancer-terms/def/car-t-cell-therapy`. Accessed: 2020-11-19.

[7] "Chimeric antigen receptor T cell." `https://https://en.wikipedia.org/wiki/Chimeric_antigen_receptor_T_cell`. Accessed: 2020-11-19.

[8] R. Twombly, "Cancer surpasses heart disease as leading cause of death for all but the very elderly," *Journal of the National Cancer Institute*, vol. 97, no. 5, pp. 330–331, 2005.

[9] M. Heron and R. N. Anderson, "Changes in the leading cause of death: recent patterns in heart disease and cancer mortality," *NCHS Data Brief*, vol. 254, 2016.

[10] E. R. Mardis and R. K. Wilson, "Cancer genome sequencing: a review," *Human Molecular Genetics*, vol. 18, no. R2, pp. R163–R168, 2009.

[11] P. Darvin, S. M. Toor, V. S. Nair, and E. Elkord, "Immune checkpoint inhibitors: recent progress and potential biomarkers," *Experimental & Molecular Medicine*, vol. 50, no. 12, pp. 1–11, 2018.

[12] P. Mauch, L. Constine, J. Greenberger, W. Knospe, J. Sullivan, J. L. Liesveld, and H. J. Deeg, "Hematopoietic stem cell compartment: acute and late effects of radiation therapy and chemotherapy," *International Journal of Radiation Oncology* Biology* Physics*, vol. 31, no. 5, pp. 1319–1339, 1995.

[13] A. M. Müller, H. E. Kohrt, S. Cha, G. Laport, J. Klein, A. E. Guardino, L. J. Johnston, K. E. Stockerl-Goldstein, E. Hanania, C. Juttner, *et al.*, "Long-term outcome of patients with metastatic breast cancer treated with high-dose chemotherapy and transplantation of purified autologous hematopoietic stem cells," *Biology of Blood and Marrow Transplantation*, vol. 18, no. 1, pp. 125–133, 2012.

[14] J. C. Doloff and D. J. Waxman, "VEGF receptor inhibitors block the ability of metronomically dosed cyclophosphamide to activate innate immunity–induced tumor regression," *Cancer Research*, vol. 72, no. 5, pp. 1103–1115, 2012.

[15] C.-S. Chen, J. C. Doloff, and D. J. Waxman, "Intermittent metronomic drug schedule is essential for activating antitumor innate immunity and tumor xenograft regression," *Neoplasia*, vol. 16, no. 1, p. 84, 2014.

[16] J. Wu and D. J. Waxman, "Metronomic cyclophosphamide eradicates large implanted GL261 gliomas by activating antitumor Cd8+ T-cell responses and immune memory," *Oncoimmunology*, vol. 4, no. 4, p. e1005521, 2015.

[17] J. Wu, M. Jordan, and D. J. Waxman, "Metronomic cyclophosphamide activation of anti-tumor immunity: tumor model, mouse host, and drug schedule dependence of gene responses and their upstream regulators," *BMC Cancer*, vol. 16, no. 1, p. 623, 2016.

[18] I. Kareva, D. J. Waxman, and G. L. Klement, "Metronomic chemotherapy: an attractive alternative to maximum tolerated dose therapy that can activate anti-tumor immunity and minimize therapeutic resistance," *Cancer Letters*, vol. 358, no. 2, pp. 100–106, 2015.

[19] D. S. Park, M. Robertson-Tessi, K. A. Luddy, P. K. Maini, M. B. Bonsall, R. A. Gatenby, and A. R. Anderson, "The goldilocks window of personalized chemotherapy: Getting the immune response just right," *Cancer Research*, vol. 79, no. 20, pp. 5302–5315, 2019.

[20] A. P. Tran, M. Ali Al-Radhawi, I. Kareva, J. Wu, D. J. Waxman, and E. D. Sontag, "Delicate balances in cancer chemotherapy: modeling immune recruitment and emergence of systemic drug resistance," *Frontiers in Immunology*, vol. 11, p. 1376, 2020.

[21] U. Alon, *An Introduction to Systems Biology: Design Principles of Biological Circuits*. Boca Raton, FL: Chapman-Hall/CRC Taylor & Francis, 2007.

[22] U. Ledzewicz, B. Amini, and H. Schättler, "Dynamics and control of a mathematical model for metronomic chemotherapy," *Mathematical Biosciences & Engineering*, vol. 12, no. 6, p. 1257, 2015.

[23] J. Ciccolini, D. Barbolosi, C. Meille, A. Lombard, C. Serdjebi, S. Giacometti, L. Padovani, E. Pasquier, and N. André, "Pharmacokinetics and pharmacodynamics-based mathematical modeling identifies an optimal protocol

for metronomic chemotherapy," *Cancer Research*, vol. 77, no. 17, pp. 4723–4733, 2017.

[24] J. M. Greene, J. L. Gevertz, and E. D. Sontag, "Mathematical approach to differentiate spontaneous and induced evolution to drug resistance during cancer treatment," *JCO Clinical Cancer Informatics*, vol. 3, pp. 1–20, 2019.

[25] B. Du and D. J. Waxman, "Medium dose intermittent cyclophosphamide induces immunogenic cell death and cancer cell autonomous type I interferon production in glioma models," *Cancer Letters*, 2019.

[26] E. Sontag, "A dynamical model of immune responses to antigen presentation predicts different regions of tumor or pathogen elimination," *Cell Systems*, vol. 4, pp. 231–241, 2017.

[27] A. Emadi, R. J. Jones, and R. A. Brodsky, "Cyclophosphamide and cancer: golden anniversary," *Nature Reviews Clinical Oncology*, vol. 6, no. 11, p. 638, 2009.

[28] M. Jordan and D. J. Waxman, "CpG-1826 immunotherapy potentiates chemotherapeutic and anti-tumor immune responses to metronomic cyclophosphamide in a preclinical glioma model," *Cancer Letters*, vol. 373, no. 1, pp. 88–96, 2016.

[29] D. Chakravarti and W. W. Wong, "Synthetic biology in cell-based cancer immunotherapy," *Trends in Biotechnology*, vol. 33, no. 8, pp. 449–461, 2015.

[30] A. A. Nielsen, B. S. Der, J. Shin, P. Vaidyanathan, V. Paralanov, E. A. Strychalski, D. Ross, D. Densmore, and C. A. Voigt, "Genetic circuit design automation," *Science*, vol. 352, no. 6281, p. aac7341, 2016.

[31] J. Shin, S. Zhang, B. S. Der, A. A. Nielsen, and C. A. Voigt, "Programming Escherichia coli to function as a digital display," *Molecular Systems Biology*, vol. 16, no. 3, 2020.

[32] M. W. Gander, J. D. Vrana, W. E. Voje, J. M. Carothers, and E. Klavins, "Digital logic circuits in yeast with CRISPR-dCas9 NOR gates," *Nature Communications*, vol. 8, no. 1, pp. 1–11, 2017.

[33] E. M. Sentovich, K. J. Singh, L. Lavagno, C. Moon, R. Murgai, A. Saldanha, H. Savoj, P. R. Stephan, R. K. Brayton, and A. Sangiovanni-Vincentelli, "SIS: A system for sequential circuit synthesis," 1992.

[34] A. Mishchenko, S. Chatterjee, and R. Brayton, "Dag-aware aig rewriting: A fresh look at combinational logic synthesis," in *2006 43rd ACM/IEEE Design Automation Conference*, pp. 532–535, IEEE, 2006.

[35] R. Brayton and A. Mishchenko, "Abc: An academic industrial-strength verification tool," in *International Conference on Computer Aided Verification*, pp. 24–40, Springer, 2010.

[36] B. A. Chabner and T. G. Roberts, "Chemotherapy and the war on cancer," *Nature Reviews Cancer*, vol. 5, no. 1, pp. 65–72, 2005.

[37] A. Gilman and F. S. Philips, "The biological actions and therapeutic applications of the b-chloroethyl amines and sulfides," *Science*, vol. 103, no. 2675, pp. 409–436, 1946.

[38] A. Gilman, "The initial clinical trial of nitrogen mustard," *The American Journal of Surgery*, vol. 105, no. 5, pp. 574–578, 1963.

[39] E. FREI III, M. KARON, R. H. LEVIN, E. J. Freireich, R. J. TAYLOR, J. HANANIAN, O. SELAWRY, J. F. HOLLAND, B. HOOGSTRATEN, I. J. WOLMAN, *et al.*, "The effectiveness of combinations of antileukemic agents in inducing and maintaining remission in children with acute leukemia," *Blood*, vol. 26, no. 5, pp. 642–656, 1965.

[40] V. T. DeVita Jr, A. A. Serpick, and P. P. Carbone, "Combination chemotherapy in the treatment of advanced hodgkin's disease," *Annals of Internal Medicine*, vol. 73, no. 6, pp. 881–895, 1970.

[41] J. H. Moxley, V. T. De Vita, K. Brace, and E. Frei, "Intensive combination chemotherapy and x-irradiation in hodgkin's disease," *Cancer Research*, vol. 27, no. 7, pp. 1258–1263, 1967.

[42] D. Hanahan and R. A. Weinberg, "The hallmarks of cancer," *Cell*, vol. 100, no. 1, pp. 57–70, 2000.

[43] D. S. Vinay, E. P. Ryan, G. Pawelec, W. H. Talib, J. Stagg, E. Elkord, T. Lichtor, W. K. Decker, R. L. Whelan, H. S. Kumara, *et al.*, "Immune evasion in cancer: Mechanistic basis and therapeutic strategies," in *Seminars in Cancer Biology*, vol. 35, pp. S185–S198, Elsevier, 2015.

[44] R. De Souza, P. Zahedi, R. M. Badame, C. Allen, and M. Piquette-Miller, "Chemotherapy dosing schedule influences drug resistance development in ovarian cancer," *Molecular Cancer Therapeutics*, vol. 10, no. 7, pp. 1289–1299, 2011.

[45] H. Zahreddine and K. Borden, "Mechanisms and insights into drug resistance in cancer," *Frontiers in Pharmacology*, vol. 4, p. 28, 2013.

[46] A. B. Shah, K. A. Rejniak, and J. L. Gevertz, "Limiting the development of anticancer drug resistance in a spatial model of micrometastases," *Mathematical Biosciences and Engineering: MBE*, vol. 13, no. 6, p. 1185, 2016.

[47] O. Kepp, A. Tesniere, F. Schlemmer, M. Michaud, L. Senovilla, L. Zitvogel, and G. Kroemer, "Immunogenic cell death modalities and their impact on cancer treatment," *Apoptosis*, vol. 14, no. 4, pp. 364–375, 2009.

[48] O. Kepp, L. Galluzzi, I. Martins, F. Schlemmer, S. Adjemian, M. Michaud, A. Q. Sukkurwala, L. Menger, L. Zitvogel, and G. Kroemer, "Molecular determinants of immunogenic cell death elicited by anticancer chemotherapy," *Cancer and Metastasis Reviews*, vol. 30, no. 1, pp. 61–69, 2011.

[49] G. Kroemer, L. Galluzzi, O. Kepp, and L. Zitvogel, "Immunogenic cell death in cancer therapy," *Annual Review of Immunology*, vol. 31, pp. 51–72, 2013.

[50] E. Vacchelli, F. Aranda, A. Eggermont, J. Galon, C. Sautès-Fridman, I. Cremer, L. Zitvogel, G. Kroemer, and L. Galluzzi, "Trial watch: Chemotherapy with immunogenic cell death inducers," *Oncoimmunology*, vol. 3, no. 3, p. e27878, 2014.

[51] L. Bracci, G. Schiavoni, A. Sistigu, and F. Belardelli, "Immune-based mechanisms of cytotoxic chemotherapy: implications for the design of novel and rationale-based combined treatments against cancer," *Cell Death and Differentiation*, vol. 21, no. 1, p. 15, 2014.

[52] L. Bezu, L. C. Gomes-da Silva, H. Dewitte, K. Breckpot, J. Fucikova, R. Spisek, L. Galluzzi, O. Kepp, and G. Kroemer, "Combinatorial strategies for the induction of immunogenic cell death," *Frontiers in Immunology*, vol. 6, p. 187, 2015.

[53] G. P. Dunn, A. T. Bruce, H. Ikeda, L. J. Old, and R. D. Schreiber, "Cancer immunoediting: from immunosurveillance to tumor escape," *Nature Immunology*, vol. 3, no. 11, p. 991, 2002.

[54] J. C. Becker, M. H. Andersen, D. Schrama, and P. thor Straten, "Immune-suppressive properties of the tumor microenvironment," *Cancer Immunology, Immunotherapy*, vol. 62, no. 7, pp. 1137–1148, 2013.

[55] R. J. Gillies and R. A. Gatenby, "Adaptive landscapes and emergent phenotypes: why do cancers have high glycolysis?," *Journal of Bioenergetics and Biomembranes*, vol. 39, no. 3, pp. 251–257, 2007.

[56] E. J. Pearce and E. L. Pearce, "Immunometabolism in 2017: driving immunity: all roads lead to metabolism," *Nature Reviews Immunology*, vol. 18, no. 2, p. 81, 2017.

[57] T. J. Curiel, G. Coukos, L. Zou, X. Alvarez, P. Cheng, P. Mottram, M. Evdemon-Hogan, J. R. Conejo-Garcia, L. Zhang, M. Burow, *et al.*, "Specific

recruitment of regulatory T cells in ovarian carcinoma fosters immune privilege and predicts reduced survival," *Nature Medicine*, vol. 10, no. 9, p. 942, 2004.

[58] I. Lee, L. Wang, A. D. Wells, M. E. Dorf, E. Ozkaynak, and W. W. Hancock, "Recruitment of Foxp3+ T regulatory cells mediating allograft tolerance depends on the CCR4 chemokine receptor," *Journal of Experimental Medicine*, vol. 201, no. 7, pp. 1037–1044, 2005.

[59] J. Yokokawa, V. Cereda, C. Remondo, J. L. Gulley, P. M. Arlen, J. Schlom, and K. Y. Tsang, "Enhanced functionality of $CD4^{+}CD25^{high}FoxP3^{+}$ regulatory T cells in the peripheral blood of patients with prostate cancer," *Clinical Cancer Research*, vol. 14, no. 4, pp. 1032–1040, 2008.

[60] T. H. Gasparoto, T. S. de Souza Malaspina, L. Benevides, E. J. F. de Melo, M. R. S. N. Costa, J. H. Damante, M. R. V. Ikoma, G. P. Garlet, K. A. Cavassani, J. S. da Silva, *et al.*, "Patients with oral squamous cell carcinoma are characterized by increased frequency of suppressive regulatory T cells in the blood and tumor microenvironment," *Cancer Immunology, Immunotherapy*, vol. 59, no. 6, pp. 819–828, 2010.

[61] J. F. Jacobs, S. Nierkens, C. G. Figdor, I. J. M. de Vries, and G. J. Adema, "Regulatory T cells in melanoma: the final hurdle towards effective immunotherapy?," *The Lancet Oncology*, vol. 13, no. 1, pp. e32–e42, 2012.

[62] C. Murdoch, M. Muthana, S. B. Coffelt, and C. E. Lewis, "The role of myeloid cells in the promotion of tumour angiogenesis," *Nature Reviews Cancer*, vol. 8, no. 8, p. 618, 2008.

[63] F. Shojaei, C. Zhong, X. Wu, L. Yu, and N. Ferrara, "Role of myeloid cells in tumor angiogenesis and growth," *Trends in Cell Biology*, vol. 18, no. 8, pp. 372–378, 2008.

[64] A. Mantovani and A. Sica, "Macrophages, innate immunity and cancer: balance, tolerance, and diversity," *Current Opinion in Immunology*, vol. 22, no. 2, pp. 231–237, 2010.

[65] L. C. van Kempen, D. J. Ruiter, G. N. van Muijen, and L. M. Coussens, "The tumor microenvironment: a critical determinant of neoplastic evolution," *European Journal of Cell Biology*, vol. 82, no. 11, pp. 539–548, 2003.

[66] A. Sica, P. Larghi, A. Mancino, L. Rubino, C. Porta, M. G. Totaro, M. Rimoldi, S. K. Biswas, P. Allavena, and A. Mantovani, "Macrophage polarization in tumour progression," in *Seminars in Cancer Biology*, vol. 18, pp. 349–355, Elsevier, 2008.

[67] J. Yang, J. Yan, and B. Liu, "Targeting VEGF/VEGFR to Modulate Antitumor Immunity," *Frontiers in Immunology*, vol. 9, p. 978, 2018.

[68] D. H. Munn, M. D. Sharma, J. R. Lee, K. G. Jhaver, T. S. Johnson, D. B. Keskin, B. Marshall, P. Chandler, S. J. Antonia, R. Burgess, *et al.*, "Potential regulatory function of human dendritic cells expressing indoleamine 2, 3-dioxygenase," *Science*, vol. 297, no. 5588, pp. 1867–1870, 2002.

[69] B. Pasche, "Role of transforming growth factor beta in cancer," *Journal of Cellular Physiology*, vol. 186, no. 2, pp. 153–168, 2001.

[70] E. M. Sotomayor, Y. X. Fu, M. Lopez-Cepero, L. Herbert, J. J. Jimenez, C. Albarracin, and D. M. Lopez, "Role of tumor-derived cytokines on the immune system of mice bearing a mammary adenocarcinoma. II. down-regulation of macrophage-mediated cytotoxicity by tumor-derived granulocyte-macrophage colony-stimulating factor.," *The Journal of Immunology*, vol. 147, no. 8, pp. 2816–2823, 1991.

[71] M. Matsuda, F. Salazar, M. Petersson, G. Masucci, J. Hansson, P. Pisa, Q.-J. Zhang, M. Masucci, and R. Kiessling, "Interleukin 10 pretreatment protects target cells from tumor-and allo-specific cytotoxic T cells and downregulates

HLA class I expression.," *Journal of Experimental Medicine*, vol. 180, no. 6, pp. 2371–2376, 1994.

[72] S. Klein, M. Jücker, H. Abts, and H. Tesch, "IL6 and IL6 receptor expression in Burkitt's lymphoma and lymphoblastoid cell lines: promotion of il6 receptor expression by ebv," *Hematological Oncology*, vol. 13, no. 3, pp. 121–130, 1995.

[73] E. Y. Lin, V. Gouon-Evans, A. V. Nguyen, and J. W. Pollard, "The macrophage growth factor CSF-1 in mammary gland development and tumor progression," *Journal of Mammary Gland Biology and Neoplasia*, vol. 7, no. 2, pp. 147–162, 2002.

[74] M. H. Lind, B. Rozell, R. P. Wallin, M. van Hogerlinden, H.-G. Ljunggren, R. Toftgård, and I. Sur, "Tumor necrosis factor receptor 1-mediated signaling is required for skin cancer development induced by NF-κB inhibition," *Proceedings of the National Academy of Sciences*, vol. 101, no. 14, pp. 4972–4977, 2004.

[75] D. I. Gabrilovich, H. L. Chen, K. R. Girgis, H. T. Cunningham, G. M. Meny, S. Nadaf, D. Kavanaugh, and D. P. Carbone, "Production of vascular endothelial growth factor by human tumors inhibits the functional maturation of dendritic cells," *Nature Medicine*, vol. 2, no. 10, p. 1096, 1996.

[76] S. Birkle, G. Zeng, L. Gao, R. Yu, and J. Aubry, "Role of tumor-associated gangliosides in cancer progression," *Biochimie*, vol. 85, no. 3-4, pp. 455–463, 2003.

[77] C. D. Mills, J. Shearer, R. Evans, and M. D. Caldwell, "Macrophage arginine metabolism and the inhibition or stimulation of cancer.," *The Journal of Immunology*, vol. 149, no. 8, pp. 2709–2714, 1992.

[78] V. Boutard, R. Havouis, B. Fouqueray, C. Philippe, J.-P. Moulinoux, and L. Baud, "Transforming growth factor-beta stimulates arginase activity in macrophages. implications for the regulation of macrophage cytotoxicity.," *The Journal of Immunology*, vol. 155, no. 4, pp. 2077–2084, 1995.

[79] Y. Xia, N. Yeddula, M. Leblanc, E. Ke, Y. Zhang, E. Oldfield, R. J. Shaw, and I. M. Verma, "Reduced cell proliferation by IKK2 depletion in a mouse lung-cancer model," *Nature Cell Biology*, vol. 14, no. 3, p. 257, 2012.

[80] G. Chen and L. A. Emens, "Chemoimmunotherapy: reengineering tumor immunity," *Cancer Immunology, Immunotherapy*, vol. 62, no. 2, pp. 203–216, 2013.

[81] J. Pol, E. Vacchelli, F. Aranda, F. Castoldi, A. Eggermont, I. Cremer, C. Sautes-Fridman, J. Fucikova, J. Galon, R. Spisek, *et al.*, "Trial watch: Immunogenic cell death inducers for anticancer chemotherapy," *Oncoimmunology*, vol. 4, no. 4, p. e1008866, 2015.

[82] M. R. Elliott, F. B. Chekeni, P. C. Trampont, E. R. Lazarowski, A. Kadl, S. F. Walk, D. Park, R. I. Woodson, M. Ostankovich, P. Sharma, *et al.*, "Nucleotides released by apoptotic cells act as a find-me signal to promote phagocytic clearance," *Nature*, vol. 461, no. 7261, p. 282, 2009.

[83] P. A. Beavis, J. Stagg, P. K. Darcy, and M. J. Smyth, "CD73: a potent suppressor of antitumor immune responses," *Trends in Immunology*, vol. 33, no. 5, pp. 231–237, 2012.

[84] G. Li, X. Liang, and M. T. Lotze, "HMGB1: the central cytokine for all lymphoid cells," *Frontiers in Immunology*, vol. 4, p. 68, 2013.

[85] H. Yanai, T. Ban, Z. Wang, M. K. Choi, T. Kawamura, H. Negishi, M. Nakasato, Y. Lu, S. Hangai, R. Koshiba, *et al.*, "HMGB proteins function as universal sentinels for nucleic-acid-mediated innate immune responses," *Nature*, vol. 462, no. 7269, p. 99, 2009.

[86] A. Sistigu, T. Yamazaki, E. Vacchelli, K. Chaba, D. P. Enot, J. Adam, I. Vitale, A. Goubar, E. E. Baracco, C. Remédios, *et al.*, "Cancer cell–autonomous contribution of type I interferon signaling to the efficacy of chemotherapy," *Nature Medicine*, vol. 20, no. 11, p. 1301, 2014.

[87] L. B. Ivashkiv and L. T. Donlin, "Regulation of type I interferon responses," *Nature Reviews Immunology*, vol. 14, no. 1, pp. 36–49, 2014.

[88] S. Makris, M. Paulsen, and C. Johansson, "Type I interferons as regulators of lung inflammation," *Frontiers in Immunology*, vol. 8, p. 259, 2017.

[89] I. Martins, O. Kepp, L. Galluzzi, L. Senovilla, F. Schlemmer, S. Adjemian, L. Menger, M. Michaud, L. Zitvogel, and G. Kroemer, "Surface-exposed calreticulin in the interaction between dying cells and phagocytes," *Annals of the New York Academy of Sciences*, vol. 1209, no. 1, pp. 77–82, 2010.

[90] R. Spisek, A. Charalambous, A. Mazumder, D. H. Vesole, S. Jagannath, and M. V. Dhodapkar, "Bortezomib enhances dendritic cell (DC)–mediated induction of immunity to human myeloma via exposure of cell surface heat shock protein 90 on dying tumor cells: therapeutic implications," *Blood*, vol. 109, no. 11, pp. 4839–4845, 2007.

[91] N. Penel, A. Adenis, and G. Bocci, "Cyclophosphamide-based metronomic chemotherapy: after 10 years of experience, where do we stand and where are we going?," *Critical Reviews in Oncology/Hematology*, vol. 82, no. 1, pp. 40–50, 2012.

[92] L. Sanchez-Perez, C. M. Suryadevara, B. D. Choi, E. A. Reap, and J. H. Sampson, "Leveraging chemotherapy-induced lymphopenia to potentiate cancer immunotherapy," *Oncoimmunology*, vol. 3, no. 7, p. e944054, 2014.

[93] M. Ahlmann and G. Hempel, "The effect of cyclophosphamide on the immune system: implications for clinical cancer therapy," *Cancer Chemotherapy and Pharmacology*, vol. 78, no. 4, pp. 661–671, 2016.

[94] L. Zitvogel, A. Tesniere, and G. Kroemer, "Cancer despite immunosurveillance: immunoselection and immunosubversion," *Nature Reviews Immunology*, vol. 6, no. 10, p. 715, 2006.

[95] F. Ghiringhelli, C. Menard, P. E. Puig, S. Ladoire, S. Roux, F. Martin, E. Solary, A. Le Cesne, L. Zitvogel, and B. Chauffert, "Metronomic cyclophosphamide regimen selectively depletes CD4+ CD25+ regulatory T cells and restores T and NK effector functions in end stage cancer patients," *Cancer Immunology, Immunotherapy*, vol. 56, no. 5, pp. 641–648, 2007.

[96] L. G. de Pillis, A. E. Radunskaya, and C. L. Wiseman, "A validated mathematical model of cell-mediated immune response to tumor growth," *Cancer Research*, vol. 65, no. 17, pp. 7950–7958, 2005.

[97] L. G. de Pillis, W. Gu, and A. E. Radunskaya, "Mixed immunotherapy and chemotherapy of tumors: modeling, applications and biological interpretations," *Journal of Theoretical Biology*, vol. 238, no. 4, pp. 841–862, 2006.

[98] H. Schättler, U. Ledzewicz, and B. Amini, "Dynamical properties of a minimally parameterized mathematical model for metronomic chemotherapy," *Journal of Mathematical Biology*, vol. 72, no. 5, pp. 1255–1280, 2016.

[99] S. Mangan and U. Alon, "Structure and function of the feed-forward loop network motif," *Proceedings of the National Academy of Sciences*, vol. 100, no. 21, pp. 11980–11985, 2003.

[100] S. Mangan, S. Itzkovitz, A. Zaslaver, and U. Alon, "The incoherent feed-forward loop accelerates the response-time of the gal system of escherichia coli," *Journal of Molecular Biology*, vol. 356, no. 5, pp. 1073–1081, 2006.

[101] R. Entus, B. Aufderheide, and H. M. Sauro, "Design and implementation of three incoherent feed-forward motif based biological concentration sensors," *Systems and Synthetic Biology*, vol. 1, no. 3, pp. 119–128, 2007.

[102] Y. Hart and U. Alon, "The utility of paradoxical components in biological circuits," *Molecular Cell*, vol. 49, pp. 213–221, Jan 2013.

[103] T. Segall-Shapiro, E. D. Sontag, and C. A. Voigt, "Engineered promoters enable constant gene expression at any copy number in bacteria," *Nature Biotechnology*, vol. 36, pp. 352–358, 2018.

[104] E. Nikolaev, A. Zloza, and E. Sontag, "Immunobiochemical reconstruction of influenza lung infection - melanoma skin cancer interactions," *Frontiers in Immunology*, vol. 10, p. Article 4, 2019.

[105] O. Shoval, L. Goentoro, Y. Hart, A. Mayo, E. Sontag, and U. Alon, "Fold change detection and scalar symmetry of sensory input fields," *Proceedings of the National Academy of Sciences*, vol. 107, pp. 15995–16000, 2010.

[106] M. Skataric, E. Nikolaev, and E. Sontag, "A fundamental limitation to fold-change detection by biological systems with multiple time scales," *IET Systems Biology*, vol. 9, pp. 1–15, 2015.

[107] M. E. De Jonge, A. D. Huitema, S. Rodenhuis, and J. H. Beijnen, "Clinical pharmacokinetics of cyclophosphamide," *Clinical Pharmacokinetics*, vol. 44, no. 11, pp. 1135–1164, 2005.

[108] U. Emmenegger, Y. Shaked, S. Man, G. Bocci, I. Spasojevic, G. Francia, A. Kouri, R. Coke, W. Cruz-Munoz, S. M. Ludeman, *et al.*, "Pharmacodynamic and pharmacokinetic study of chronic low-dose metronomic cyclophosphamide therapy in mice," *Molecular Cancer Therapeutics*, vol. 6, no. 8, pp. 2280–2289, 2007.

[109] C. Banissi, F. Ghiringhelli, L. Chen, and A. F. Carpentier, "Treg depletion with a low-dose metronomic temozolomide regimen in a rat glioma model," *Cancer Immunology, Immunotherapy*, vol. 58, no. 10, pp. 1627–1634, 2009.

[110] D. Generali, G. Bates, A. Berruti, M. P. Brizzi, L. Campo, S. Bonardi, A. Bersiga, G. Allevi, M. Milani, S. Aguggini, *et al.*, "Immunomodulation of FOXP3+ regulatory t cells by the aromatase inhibitor letrozole in breast cancer patients," *Clinical Cancer Research*, vol. 15, no. 3, pp. 1046–1051, 2009.

[111] H. Hong, A. Ovchinnikov, G. Pogudin, and C. Yap, "Sian: software for structural identifiability analysis of ode models," *Bioinformatics*, vol. 35, no. 16, pp. 2873–2874, 2019.

[112] R. Cukier, C. Fortuin, K. E. Shuler, A. Petschek, and J. Schaibly, "Study of the sensitivity of coupled reaction systems to uncertainties in rate coefficients. i theory," *The Journal of chemical physics*, vol. 59, no. 8, pp. 3873–3878, 1973.

[113] M. A. Al-Radhawi, A. P. Tran, E. A. Ernst, T. Chen, C. A. Voigt, and E. D. Sontag, "Distributed implementation of boolean functions by transcriptional synthetic circuits," *ACS Synthetic Biology*, vol. 9, no. 8, pp. 2172–2187, 2020. PMID: 32589837.

[114] H. H. McAdams and L. Shapiro, "Circuit simulation of genetic networks," *Science*, vol. 269, no. 5224, pp. 650–656, 1995.

[115] P.-F. Xia, H. Ling, J. L. Foo, and M. W. Chang, "Synthetic genetic circuits for programmable biological functionalities," *Biotechnology Advances*, 2019.

[116] A. S. Khalil and J. J. Collins, "Synthetic biology: applications come of age," *Nature Reviews Genetics*, vol. 11, no. 5, pp. 367–379, 2010.

[117] J. H. Esensten, J. A. Bluestone, and W. A. Lim, "Engineering therapeutic T cells: from synthetic biology to clinical trials," *Annual Review of Pathology: Mechanisms of Disease*, vol. 12, pp. 305–330, 2017.

[118] M. L. Davila, I. Riviere, X. Wang, S. Bartido, J. Park, K. Curran, S. S. Chung, J. Stefanski, O. Borquez-Ojeda, M. Olszewska, *et al.*, "Efficacy and toxicity management of 19-28z CAR T cell therapy in B cell acute lymphoblastic leukemia," *Science Translational Medicine*, vol. 6, no. 224, pp. 224ra25–224ra25, 2014.

[119] D. W. Lee, J. N. Kochenderfer, M. Stetler-Stevenson, Y. K. Cui, C. Delbrook, S. A. Feldman, T. J. Fry, R. Orentas, M. Sabatino, N. N. Shah, *et al.*, "T cells

expressing CD19 chimeric antigen receptors for acute lymphoblastic leukaemia in children and young adults: a phase 1 dose-escalation trial," *The Lancet*, vol. 385, no. 9967, pp. 517–528, 2015.

[120] P. Narayanan, N. Lapteva, M. Seethammagari, J. M. Levitt, K. M. Slawin, and D. M. Spencer, "A composite MyD88/CD40 switch synergistically activates mouse and human dendritic cells for enhanced antitumor efficacy," *The Journal of Clinical Investigation*, vol. 121, no. 4, pp. 1524–1534, 2011.

[121] K. Kudo, C. Imai, P. Lorenzini, T. Kamiya, K. Kono, A. M. Davidoff, W. J. Chng, and D. Campana, "T lymphocytes expressing a CD16 signaling receptor exert antibody-dependent cancer cell killing," *Cancer Research*, vol. 74, no. 1, pp. 93–103, 2014.

[122] J. S. Park, B. Rhau, A. Hermann, K. A. McNally, C. Zhou, D. Gong, O. D. Weiner, B. R. Conklin, J. Onuffer, and W. A. Lim, "Synthetic control of mammalian-cell motility by engineering chemotaxis to an orthogonal bioinert chemical signal," *Proceedings of the National Academy of Sciences*, vol. 111, no. 16, pp. 5896–5901, 2014.

[123] B. S. Jones, L. S. Lamb, F. Goldman, and A. Di Stasi, "Improving the safety of cell therapy products by suicide gene transfer," *Frontiers in Pharmacology*, vol. 5, p. 254, 2014.

[124] V. Marin, E. Cribioli, B. Philip, S. Tettamanti, I. Pizzitola, A. Biondi, E. Biagi, and M. Pule, "Comparison of different suicide-gene strategies for the safety improvement of genetically manipulated T cells," *Human Gene Therapy Methods*, vol. 23, no. 6, pp. 376–386, 2012.

[125] K. C. Straathof, M. A. Pule, P. Yotnda, G. Dotti, E. F. Vanin, M. K. Brenner, H. E. Heslop, D. M. Spencer, and C. M. Rooney, "An inducible caspase 9 safety switch for t-cell therapy," *Blood*, vol. 105, no. 11, pp. 4247–4254, 2005.

[126] L. E. Budde, C. Berger, Y. Lin, J. Wang, X. Lin, S. E. Frayo, S. A. Brouns, D. M. Spencer, B. G. Till, M. C. Jensen, *et al.*, "Combining a CD20 chimeric antigen receptor and an inducible caspase 9 suicide switch to improve the efficacy and safety of T cell adoptive immunotherapy for lymphoma," *PLoS One*, vol. 8, no. 12, p. e82742, 2013.

[127] A. Di Stasi, S.-K. Tey, G. Dotti, Y. Fujita, A. Kennedy-Nasser, C. Martinez, K. Straathof, E. Liu, A. G. Durett, B. Grilley, *et al.*, "Inducible apoptosis as a safety switch for adoptive cell therapy," *The New England Journal of Medicine*, vol. 365, pp. 1673–1683, 2011.

[128] J. H. Cho, J. J. Collins, and W. W. Wong, "Universal chimeric antigen receptors for multiplexed and logical control of T cell responses," *Cell*, vol. 173, no. 6, pp. 1426–1438, 2018.

[129] E. W. Weber, M. V. Maus, and C. L. Mackall, "The emerging landscape of immune cell therapies," *Cell*, vol. 181, no. 1, pp. 46–62, 2020.

[130] J. A. Brophy and C. A. Voigt, "Principles of genetic circuit design," *Nature Methods*, vol. 11, no. 5, p. 508, 2014.

[131] S. Itzkovitz, T. Tlusty, and U. Alon, "Coding limits on the number of transcription factors," *BMC Genomics*, vol. 7, no. 1, p. 239, 2006.

[132] A. Garg, J. J. Lohmueller, P. A. Silver, and T. Z. Armel, "Engineering synthetic TAL effectors with orthogonal target sites," *Nucleic Acids Research*, vol. 40, no. 15, pp. 7584–7595, 2012.

[133] B. C. Stanton, A. A. Nielsen, A. Tamsir, K. Clancy, T. Peterson, and C. A. Voigt, "Genomic mining of prokaryotic repressors for orthogonal logic gates," *Nature Chemical Biology*, vol. 10, no. 2, p. 99, 2014.

[134] L. S. Qi, M. H. Larson, L. A. Gilbert, J. A. Doudna, J. S. Weissman, A. P. Arkin, and W. A. Lim, "Repurposing CRISPR as an RNA-guided platform for

sequence-specific control of gene expression," *Cell*, vol. 152, no. 5, pp. 1173–1183, 2013.

[135] L. A. Gilbert, M. H. Larson, L. Morsut, Z. Liu, G. A. Brar, S. E. Torres, N. Stern-Ginossar, O. Brandman, E. H. Whitehead, J. A. Doudna, *et al.*, "CRISPR-mediated modular RNA-guided regulation of transcription in eukaryotes," *Cell*, vol. 154, no. 2, pp. 442–451, 2013.

[136] A. A. Nielsen and C. A. Voigt, "Multi-input CRISPR/Cas genetic circuits that interface host regulatory networks," *Molecular Systems Biology*, vol. 10, no. 11, 2014.

[137] J. M. Rock, F. F. Hopkins, A. Chavez, M. Diallo, M. R. Chase, E. R. Gerrick, J. R. Pritchard, G. M. Church, E. J. Rubin, C. M. Sassetti, *et al.*, "Programmable transcriptional repression in mycobacteria using an orthogonal CRISPR interference platform," *Nature Microbiology*, vol. 2, no. 4, pp. 1–9, 2017.

[138] L. Cui, A. Vigouroux, F. Rousset, H. Varet, V. Khanna, and D. Bikard, "A CRISPRi screen in E. coli reveals sequence-specific toxicity of dCas9," *Nature Communications*, vol. 9, no. 1, pp. 1–10, 2018.

[139] S. Zhang and C. A. Voigt, "Engineered dCas9 with reduced toxicity in bacteria: implications for genetic circuit design," *Nucleic Acids Research*, vol. 46, no. 20, pp. 11115–11125, 2018.

[140] S. Cho, D. Choe, E. Lee, S. C. Kim, B. Palsson, and B.-K. Cho, "High-level dCas9 expression induces abnormal cell morphology in Escherichia coli," *ACS Synthetic Biology*, vol. 7, no. 4, pp. 1085–1094, 2018.

[141] B. M. Markus, G. W. Bell, H. A. Lorenzi, and S. Lourido, "Optimizing systems for Cas9 expression in Toxoplasma gondii," *mSphere*, vol. 4, no. 3, pp. e00386–19, 2019.

[142] J. T. Sexton, *Multiplexing cell-cell communication*. PhD thesis, Rice University, 2019.

[143] L. B. Andrews, A. A. Nielsen, and C. A. Voigt, "Cellular checkpoint control using programmable sequential logic," *Science*, vol. 361, no. 6408, p. eaap8987, 2018.

[144] J. Santos-Moreno, E. Tasiudi, J. Stelling, and Y. Schaerli, "Multistable and dynamic crispri-based synthetic circuits," *Nature Communications*, vol. 11, no. 1, pp. 1–8, 2020.

[145] J. Macía, F. Posas, and R. V. Solé, "Distributed computation: the new wave of synthetic biology devices," *Trends in Biotechnology*, vol. 30, no. 6, pp. 342–349, 2012.

[146] Y. Xiang, N. Dalchau, and B. Wang, "Scaling up genetic circuit design for cellular computing: advances and prospects," *Natural Computing*, vol. 17, no. 4, pp. 833–853, 2018.

[147] S. Basu, Y. Gerchman, C. H. Collins, F. H. Arnold, and R. Weiss, "A synthetic multicellular system for programmed pattern formation," *Nature*, vol. 434, no. 7037, pp. 1130–1134, 2005.

[148] S. Wang, G. F. Payne, and W. E. Bentley, "Quorum sensing communication: Molecularly connecting cells, their neighbors, and even devices," *Annual Review of Chemical and Biomolecular Engineering*, vol. 11, 2020.

[149] A. Goñi-Moreno, M. Amos, and F. de la Cruz, "Multicellular computing using conjugation for wiring," *PLoS One*, vol. 8, no. 6, 2013.

[150] A. Tamsir, J. J. Tabor, and C. A. Voigt, "Robust multicellular computing using genetically encoded NOR gates and chemical 'wires'," *Nature*, vol. 469, no. 7329, pp. 212–215, 2011.

[151] W. Ji, H. Shi, H. Zhang, R. Sun, J. Xi, D. Wen, J. Feng, Y. Chen, X. Qin, Y. Ma, *et al.*, "A formalized design process for bacterial consortia that perform logic computing," *PLoS One*, vol. 8, no. 2, 2013.

[152] J. Macia, R. Manzoni, N. Conde, A. Urrios, E. de Nadal, R. Solé, and F. Posas, "Implementation of complex biological logic circuits using spatially distributed multicellular consortia," *PLoS Computational Biology*, vol. 12, no. 2, 2016.

[153] S. Guiziou, F. Ulliana, V. Moreau, M. Leclere, and J. Bonnet, "An automated design framework for multicellular recombinase logic," *ACS Synthetic Biology*, vol. 7, no. 5, pp. 1406–1412, 2018.

[154] D. Ausländer, S. Ausländer, X. Pierrat, L. Hellmann, L. Rachid, and M. Fussenegger, "Programmable full-adder computations in communicating three-dimensional cell cultures," *Nature Methods*, vol. 15, no. 1, p. 57, 2018.

[155] M. R. Regueira, J. D. García, and A. R.-P. Aradas, "The multicellular incoherent feedforward loop motif generates spatial patterns," *bioRxiv*, p. 579342, 2019.

[156] R. Zhang, H. Goetz, J. Melendez-Alvarez, J. Li, T. Ding, X. Wang, and X.-J. Tian, "Winner-takes-all resource competition redirects cascading cell fate transitions," *bioRxiv*, 2020.

[157] A. L. Schaefer, E. Greenberg, C. M. Oliver, Y. Oda, J. J. Huang, G. Bittan-Banin, C. M. Peres, S. Schmidt, K. Juhaszova, J. R. Sufrin, *et al.*, "A new class of homoserine lactone quorum-sensing signals," *Nature*, vol. 454, no. 7204, pp. 595–599, 2008.

[158] J. P. Pearson, L. Passador, B. H. Iglewski, and E. Greenberg, "A second N-acylhomoserine lactone signal produced by Pseudomonas aeruginosa.," *Proceedings of the National Academy of Sciences*, vol. 92, no. 5, pp. 1490–1494, 1995.

[159] M. M. Mano, *Digital Design*. Pearson Prentice Hall, 2013.

[160] B. W. Kernighan and S. Lin, "An efficient heuristic procedure for partitioning graphs," *The Bell System Technical Journal*, vol. 49, no. 2, pp. 291–307, 1970.

[161] D. A. Papa and I. L. Markov, "Hypergraph partitioning and clustering," in *Approximation Algorithms and Metaheuristics* (T. F. Gonzalez, ed.), Chapman & Hall/CRC, 2007.

[162] R. Brayton and A. Mishchenko, "ABC: An academic industrial-strength verification tool," in *Proceedings of the International Conference on Computer Aided Verification*, pp. 24–40, Springer, 2010.

[163] R. K. Brayton, G. D. Hachtel, C. McMullen, and A. Sangiovanni-Vincentelli, *Logic minimization algorithms for VLSI synthesis*. Springer, 1984.

[164] A. A. Nielsen, *Biomolecular and computational frameworks for genetic circuit design*. PhD thesis, Massachusetts Institute of Technology, 2017.

[165] R. K. Brayton, G. D. Hachtel, C. McMullen, and A. Sangiovanni-Vincentelli, *Logic minimization algorithms for VLSI synthesis*, vol. 2. Springer Science & Business Media, 1984.

[166] R. K. Brayton, G. D. Hachtel, and A. L. Sangiovanni-Vincentelli, "Multilevel logic synthesis," *Proceedings of the IEEE*, vol. 78, no. 2, pp. 264–300, 1990.

[167] R. L. Rudell and A. Sangiovanni-Vincentelli, "Multiple-valued minimization for pla optimization," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 6, no. 5, pp. 727–750, 1987.

[168] E. A. Ernst, *Optimal combinational multi-level logic synthesis*. PhD thesis, University of Michigan, 2009.

[169] R. L. Ashenhurst, "The decomposition of switching functions," in *Proceedings of an International Symposium on the Theory of Switching, April 1957*, 1957.

[170] H. A. Curtis, "A generalized tree circuit," *Journal of the ACM (JACM)*, vol. 8, no. 4, pp. 484–496, 1961.

[171] R. M. Karp, F. McFarlin, J. P. Roth, and J. Wilts, "A computer program for the synthesis of combinational switching circuits," in *2nd Annual Symposium on Switching Circuit Theory and Logical Design (SWCT 1961)*, pp. 182–194, IEEE, 1961.

[172] J. P. Roth, "Minimization over boolean trees," *IBM Journal of Research and Development*, vol. 4, no. 5, pp. 543–558, 1960.

[173] J. P. Roth and R. M. Karp, "Minimization over boolean graphs," *IBM journal of Research and Development*, vol. 6, no. 2, pp. 227–238, 1962.

[174] L. Hellerman, "A catalog of three-variable or-invert and and-invert logical circuits," *IEEE Transactions on Electronic Computers*, no. 3, pp. 198–223, 1963.

[175] R. A. Smith, "Minimal three-variable nor and nand logic circuits," *IEEE Transactions on Electronic Computers*, no. 1, pp. 79–81, 1965.

[176] R. Drechsler and W. Günther, "Exact circuit synthesis," in *In Int'l Workshop on Logic Synth*, Citeseer, 1998.

[177] S. Muroga and T. Ibaraki, "Design of optimal switching networks by integer programming," *IEEE Transactions on Computers*, vol. 100, no. 6, pp. 573–582, 1972.

[178] E. S. Davidson, "An algorithm for nand decomposition under network constraints," *IEEE Transactions on Computers*, vol. 100, no. 12, pp. 1098–1109, 1969.

[179] T. Nakagawa, H. Lai, and S. Muroga, "Design algorithm of optimal logic networks by the branch-and-bound approach.," *J. COMP. AIDED VLSI DES.*, vol. 1, no. 2, pp. 203–231, 1989.

[180] J. A. Darringer, W. H. Joyner, C. L. Berman, and L. Trevillyan, "Logic synthesis through local transformations," *IBM Journal of Research and Development*, vol. 25, no. 4, pp. 272–280, 1981.

[181] A. Vincentelli, A. Wang, R. Brayton, and R. Rudell, "Mis: Multiple level logic optimization system," *IEEE Transactions on Computer Aided Design of Integrated Circuits and Systems*, 1987.

[182] K. Bartlett, W. Cohen, A. De Geus, and G. Hachtel, "Synthesis and optimization of multilevel logic under timing constraints," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 5, no. 4, pp. 582–596, 1986.

[183] K. A. Bartlett, R. K. Brayton, G. D. Hachtel, R. M. Jacoby, C. R. Morrison, R. L. Rudell, A. Sangiovanni-Vincentelli, and A. Wang, "Multi-level logic minimization using implicit don't cares," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 7, no. 6, pp. 723–740, 1988.

[184] D. Brand, "Logic synthesis," in *Design Systems for VLSI Circuits: Logic Synthesis and Silicon Compilation*, pp. 301–326, Martinus Nijhoff, 1987.

[185] G. Hachtel, M. Lightner, K. Bartlett, D. Bostwick, R. Jacoby, P. Moceyunas, C. Morrison, X. Du, and E. Schwarz, "Bold: The boulder optimal logic design system," in *Proceedings of the Twenty-Second Annual Hawaii International Conference on System Sciences. Volume 1: Architecture Track*, vol. 1, pp. 59–60, IEEE Computer Society, 1989.

[186] R. K. Brayton, *Algorithms for multi-level logic synthesis and optimization*. IBM Thomas J. Watson Research Center, 1986.

[187] L. Berman and L. Trevillyan, "A global approach to circuit size reduction," in *Proceedings of the fifth MIT conference on Advanced research in VLSI*, pp. 203–214, 1988.

[188] C. L. Berman and L. H. Trevillyan, "Improved logic optimization using global-flow analysis," in *The Best of ICCAD*, pp. 217–225, Springer, 2003.

[189] E. Sentovich, K. Singh, L. Lavagno, C. Moon, R. Murgai, A. Saldanha, H. Savoj, P. Stephan, R. K. Brayton, and A. L. Sangiovanni-Vincentelli, "SIS: A system

for sequential circuit synthesis," Tech. Rep. UCB/ERL M92/41, EECS Department, University of California, Berkeley, May 1992.

[190] E. M. Sentovich, K. J. Singh, C. Moon, H. Savoj, R. K. Brayton, and A. Sangiovanni-Vincentelli, "Sequential circuit design using synthesis and optimization," in *Proceedings 1992 IEEE International Conference on Computer Design: VLSI in Computers & Processors*, pp. 328–333, IEEE, 1992.

[191] A. Kuehlmann, V. Paruthi, F. Krohm, and M. K. Ganai, "Robust boolean reasoning for equivalence checking and functional property verification," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 21, no. 12, pp. 1377–1394, 2002.

[192] Y. Crama and P. L. Hammer, *Boolean Functions: Theory, Algorithms, and Applications*. Cambridge University Press, 2011.

[193] T. Murata, "Petri nets: Properties, analysis and applications," *Proceedings of the IEEE*, vol. 77, no. 4, pp. 541–580, 1989.

[194] Y. Chen, S. Zhang, E. M. Young, T. S. Jones, D. Densmore, and C. A. Voigt, "Genetic circuit design automation for yeast," *Nature Microbiology*, vol. 5, no. 11, pp. 1349–1360, 2020.